Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Jednoduché komunikační zařízení s využitím technologie Bluetooth 4.0

Simple Communication Device with Bluetooth 4.0 Support

2019 **Ondřej Freisler**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Ondřej Freisler**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Jednoduché komunikační zařízení s využitím technologie Bluetooth 4.0
Simple Communication Device with Bluetooth 4.0 Support

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem bakalářské práce je navrhnout jednoduché komunikační zařízení s využitím Arduino a technologie bluetooth 4.0. Obsah bakalářské práce musí splňovat následující body zadání:

1. Teoretický rozbor vývoje aplikací pro Arduino, technologie Bluetooth 4.0.
2. Průzkum trhu v oblasti Arduino řešení, volba správného modulu pro Bluetooth 4.0.
3. Návrh řešení pro propojení Arduino s Bluetooth 4.0 modulem k mobilním zařízením s operačním systémem iOS a Android.
4. Nastavení Bluetooth 4.0 modulu pro neustálé vysílání a příjem tzv. "beaconů".
5. Otestování dosahu a náročností na elektrickou energii.
6. Zpracování dokumentace navrženého řešení.

Seznam doporučené odborné literatury:

[1] Townsend K., Cufí C., Akiba, Davidson R. Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking. O'Reilly Media 2014
[2] Allan A., Coleman D., Mistry S. Make: Bluetooth: Bluetooth LE Projects with Arduino, Raspberry Pi, and Smartphones (Make : Technology on Your Time). Maker Media 2015

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Lukáš Kapičák**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2019

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky 1D16 OSTRAVA

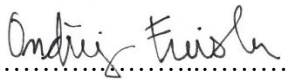
prof. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou/diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: *16. dubna 2019*



……

podpis studenta

Poděkování

Rád bych chtěl poděkovat vedoucímu bakalářské práce Ing. Lukáši Kapičákovi za odbornou pomoc a konzultaci při vytváření této bakalářské práce.

Abstrakt

Cílem této bakalářské práce bylo podat základní informace o platformě Arduino, a to jak vzniklo, jaké je jeho využití v informačních a komunikačních technologiích, jaké je jeho zastoupení na trhu a speciálně se zde pojednává o jeho propojení s technologií Bluetooth, která v dnešní době patří mezi nejznámější a nejvíce využívané technologie pro přenos dat. Po kapitole, ve které byla podrobně zpracována technologie Bluetooth, následuje praktická část, věnující se vývoji komunikačního zařízení, a to včetně grafického vysvětlení principu jeho funkce. Toto zařízení slouží k vysílání a získávání uživatelských dat skrze bezdrátovou technologii Bluetooth. Podstatná část práce se dále věnuje vývoji tohoto zařízení ve vývojových prostředích Android Studio a Arduino IDE. Zařízení bylo také testováno v souvislosti s dosahem a náročností na elektrickou energii. V závěrečné části byly uvedeny výsledky z testování.

Klíčová slova

Bluetooth; Arduino; Android; Java; přenos dat; bezdrátová komunikace; uživatelská data; vysílač; skener; přijímač

Abstract

The aim of this bachelor thesis was to give basic information about Arduino platform, how it was created, what is its usage in information and communication technologies, what is its market presence and specially it is about its connection with the Bluetooth technology which belongs to the most famous and widely used data transfer technologies. After the chapter in which the Bluetooth technology was processed in detail follows the practical part, which is dedicated to the development of communication device, including a graphical explanation of its function. The device is used to transfer and retrieve user data through Bluetooth wireless technology. A substantial part of the thesis is dedicated to the development of this device in Android Studio and Arduino IDE development environments. The device has also been tested for range and power requirements. The results of the testing were presented in the final part.

Key words

Bluetooth; Arduino; Android; Java; data transmission; wireless communication; user data; transmitter; scanner; receiver

Obsah

Seznam obrázků

Seznam tabulek

Úvod

S technologickými pokroky dnešní moderní doby rostou i možnosti rozvoje v oblasti informačních a telekomunikačních zařízení. Lidé využívají svá mobilní zařízení k ovládání stále lepších, důmyslnějších a propracovanějších mobilních aplikací, které jim v mnohém usnadňují život. Žijeme v době, kdy už není nepřekonatelný problém si přes internet zajistit či koupit produkt vyrobený kdekoliv ve světě. Shrnutím těchto faktů se dostáváme ke vzájemné kombinaci hojně využívané a všemi známé bezdrátové komunikační technologie Bluetooth a platformy Arduino, která spojuje odvětví, jako jsou programování a elektronika. Díky existenci těchto technologií a velké rozmanitosti a dostupnosti vývojových prostředí jsme schopni vymyslet a sestrojit zařízení, která pomáhají lidem usnadnit ať už zavedené principy, které by se daly vyřešit lépe a rychleji, tak také každodenní činnosti, při kterých lidé využívají moderní technologie.

Tato bakalářská práce, jak již její oficiální název napovídá, se zabývá právě technologií Bluetooth a platformou Arduino, jejíchž vzájemným propojením vznikají nové možnosti komunikace. Příkladem využití této komunikace je ono jednoduché komunikační zařízení, vyvíjené za pomoci vývojových prostředí Android Studio a Arduino IDE. Jedním z úkolů bylo také toto zařízení vytvořit, čehož bylo v rámci práce dosaženo. Zařízení bylo nakonec zdokumentováno, otestováno a byl proveden závěr, shrnující tuto práci.

1 Analýza nízkonákladové platformy Arduino

Arduino [29] je open-source elektronická platforma, vycházející z uživatelsky jednoduchého hardware i software. Možnost setkat se s ním mají lidé, zajímající se o programování jednoduchých přístrojů či zařízení. Mezi tyto lidi nepatří jen programátoři, studenti, umělci, designéři, ale třeba i domácí kutilové, kteří si chtějí doma vytvořit nějaké chytré zařízení. S platformou Arduino je možno se také setkat na školách, zaměřených na informatiku či telekomunikační technologie, a to v rámci předmětů, zabývajících se programováním a hardwarem. LED diody, senzory, displeje, světla, roboti, motory, automaty, virtuální realita, zabezpečovací systémy, mechanické úkony, bezdrátové technologie – to je jen krátký výčet toho, kde by se dalo Arduino využít. Arduino deska získá údaje od různých senzorů a snímačů. Na základě těchto údajů ovládá konkrétní výstupy (např. zapne motor, rozsvítí světlo, atd.) [1][2].

Obrázek 1: Arduino logo [22]

1.1 Historie platformy Arduino

Projekt Arduino se úplně poprvé objevil v Interakčním designovém institutu (*IIDI*) [30] v italském městě Ivrea, kde jistý kolumbijský student Hernando Barragan [4] vytvořil jako svou diplomovou práci vývojovou platformu Wiring, která měla uživatelsky přívětivé a jednoduché vývojové prostředí (*IDE – Integrated Development Environment*). Ukázalo se, že tento projekt má naději být v budoucnu úspěšný. Barraganův vedoucí práce Massimo Banzi, který pracoval jako softwarový inženýr, se musel do té doby spoléhat na mikrokontroléry BASIC Stamp, vyvíjených firmou Parallax. Ty ovšem měly podle Banziho nedostatky, a to nedostatečný výpočetní výkon, vysokou cenu a taky nedostatečná kompatibilita s různými operačními systémy. S odkazem na Wiring nakonec vytvořil Banzi s pomocí několika kolegů Arduino, které bylo levné, jednoduché a snadno použitelné [3][4].

Zajímavostí je, odkud se vlastně název Arduino vzal. Ve městě Ivrea vládl zhruba před tisíciletím král Arduin a je po něm ve městě pojmenována hospoda "Bar Di Re Arduino". Právě to je ono místo, kde se zakladatelé Arduina scházeli a kde tato platforma má své kořeny [3].

V roce 2005 vyšel první prototyp. Banzi a jeho tým ovšem měli za cíl vyvinout lehce dostupnou a levnou platformu, a proto se rozhodli, že bude lepší udělat Arduino open-source, aby bylo dostupné co nejvíce lidem (open-source byl do té doby většinou využíván u software, nikoliv hardware). Důležitý faktor v tomto rozhodnutí hrál také fakt, že institut IIDI již neměl další prostředky na samostatný vývoj.

Aby se tento přechod na open-source mohl uskutečnit, musel Arduino tým najít vhodnou licenci. Jako ta pravá se ukázala být licence Creative Commons, která byla využívána v kultuře a hudbě. Banzi v souvislosti s přechodem Arduino platformy na open-source uvedl, že samotný hardware se dá vlastně považovat za část kultury, která by měla být sdílena mezi ostatní lidi.

Dalším krokem bylo vytvoření samotné první vývojové desky s mikroprocesorem. Tým chtěl, aby byla deska v porovnání s konkurencí výjimečná a také aby dobře vypadala. Proto byla mezi ostatními vývojovými deskami, které se vyráběly v drtivé většině v zelených barvách, zvolena barva modrá. Ušetřeno nebylo ani na výrobě vstupních a výstupních pinů, neboť i ty měly zaručovat reprezentativní kvalitu. Dalším důležitým krokem k pozdějšímu úspěchu bylo to, že deska byla rychle použitelná, a to díky využití technologie Plug and play, kdy stačilo pouhé využití USB kabelu a jeho propojení mezi deskou a počítačem.

Tým se nakonec rozhodl tuto filozofii otestovat a rozdali studentům IDII 300 prázdných obvodových desek s online návodem a s tím, že mají něco vytvořit. Vzešlo z toho mnoho skvělých projektů a takovýchto desek chtělo postupem času více a více lidí. Slovo Arduino se rázem stalo na internetu populární [3][4].

1.2 Proč se využívá právě Arduino?

Arduino se dá využít v nejrůznějších projektech a aplikacích a mezi jeho základní přednosti patří následující:

- **Open-source a jeho rozšiřitelnost** – Arduino software je publikován jako nástroj s otevřeným zdrojovým kódem, který je snadno rozšiřitelný.
- **Jednoduché programovací rozhraní** – Arduino je jednoduché a srozumitelné pro začátečníky, avšak dostatečně flexibilní i pro zkušené uživatele.
- **Platformní nezávislost** – Vývojové prostředí (*IDE*) Arduina je provozováno jak na operačních systémech Windows, Linux, tak i Mac OS X, kdežto většina ostatních mikrokontrolérů je limitována pouze na operační systém Windows.
- **Cena a dostupnost** – Platforma Arduino je v dnešní moderní době na trhu zastoupena dostatečně a není problém najít si model, který potřebujeme. Existuje totiž mnoho internetových obchodů, které nabízejí nejrůznější verze desek a všemožných doplňků, takže každý si najde to své. Samozřejmostí je také velká podpora různých návodů a tutoriálů. Cena je další pozitivum, neboť ve srovnání s ostatními mikrokontroléry si Arduino vede velmi dobře. Nejlevnější Arduino si dokonce můžete sestavit ručně.
- **Plug and play** – Jak již bylo zmíněno v předešlé podkapitole, využití této technologie velmi usnadňuje zapojení a spárování samotného zařízení s počítačem [1].

1.3 Hardware

Základní Arduino deska je velmi jednoduchá a dá se vyrobit svépomocí nebo pospojováním součástek na nepájivém poli. Mezi jeho základní součástky patří mikroprocesor, krystal, 5V zdroj napájení a převodník pro komunikaci s počítačem.

Podle funkce, která je od našeho projektu vyžadována, je možností si zvolit z širokého výběru rozšiřujících desek pro Arduino. Tyto rozšiřující desky se nazývají Arduino Shieldy a jsou snadno

připojitelné. V dnešní době jich existuje velké množství, a navíc ke každému je většinou volně dostupná i knihovna pro okamžité použití.

O detailním popisu Arduino vývojové desky je v této práci zmínka v kapitole Průzkum trhu v oblasti Arduino řešení [2][5].

1.4 Software

Pro vývoj aplikací pro Arduino se využívá vývojové prostředí Arduino IDE. To je samozřejmě dostupné jak ve verzích pro Windows, Linux tak Mac OS X. Jedná se o otevřené vývojové prostředí, které programátorovi zjednodušuje psaní kódu díky své jednoduchosti. Toto prostředí je založeno na otevřeném projektu Processing (založeném na Javě), což je nástroj, využívaný pro animaci, vizualizaci a řízení hardware přímo z prostředí osobního počítače [5].

1.4.1 Programovací jazyk

Využívá se zde jazyka Arduino Programmable Language, který je založen na již dříve zmíněném jazyce Wiring. Ten zase vychází z jazyka C/C++. Samotný program, ve kterém je psán zdrojový kód, se obecně nazývá **sketch**. Tento sketch je nahrán na samotnou Arduino desku, na které následně pracuje. Pokud jde o strukturu kódu, musí v něm být obsaženy dvě hlavní předdefinované funkce návratového typu void, které nevolá programátor, ale samo Arduino:

- **setup**– Slouží k definici vstupů a výstupů, volá se pouze jednou – na začátku sketche (popř. při restartu Arduino desky). Inicializují se zde také knihovny.
- **loop** – Po opuštění funkce setup program volá tuto funkci neustále dokola, dokud program nevypneme. Naše zařízení reaguje na kód, napsaný v této funkci [5][11].

Obrázek 2: Popis vývojového prostředí Arduino IDE

2 Bluetooth

Bluetooth je označení pro bezdrátový komunikační standard, využívající bezdrátovou radiovou komunikaci na krátké vzdálenosti, sloužící k propojení elektronických zařízení. Zpočátku byl tento standard zaměřen pouze na mobilní telefony nebo na propojení mobilního telefonu a stolního počítače, ovšem postupem času se Bluetooth technologie dostala do různých zařízení, které využívají vzájemnou komunikaci k přenosu dat. V současné době je využití Bluetooth technologie tak rozsáhlé, že se s ním setkáme snad ve všech odvětvích, vyžadujících práci s informačními technologiemi. Označení pro tento standard nese název IEEE 802.15.1 (*Institute of Electrical and Electronics Engineers*) a vznikl jako náhrada kabelového propojení mezi zařízeními komunikační a výpočetní techniky [12].

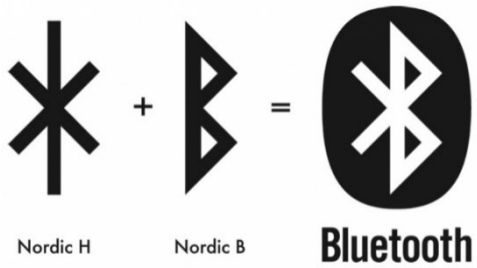
Mezi jeho hlavní přednosti lze uvést snadnou implementaci, přijatelnou cenu a nízkou náročnost na napájení. Jako další pozitivní vlastnost by se dalo uvést, že zařízení, komunikující mezi sebou přes tuto technologii nemusí být navzájem přímo viditelné.

2.1 Počátky vzniku technologie Bluetooth

S myšlenkou nahrazení kabelového propojení dvou mobilních telefonů nebo telefonu a stolního počítače na krátkou vzdálenost (WPAN – Wireless Personal Area Network) přišla švédská společnost Ericsson. Roku 1998 založila pětice firem (Ericsson, IBM, Intel, Nokia a Toshiba) sdružení Bluetooth SIG (*Special Interest Group*). Psal se rok 1999, kdy vyšla první specifikace tohoto nového standardu [12][13].

2.1.1 Historie vzniku názvu "Bluetooth"

Název Bluetooth (česky *"Modrý zub"*) vychází až z daleké vikingské historie. Dánský a norský král Harald Blåtand, žijící v 10. století, proslul svými diplomatickými dovednostmi. Přesvědčil totiž válčící kmeny uzavřít mír a začít mezi sebou komunikovat. Shodou okolností jeden z vývojářů sdružení Bluetooth SIG četl na jednom meetingu knihu o vikingské historii, ve které se nechal právě inspirovat tímto králem. Po překladu jeho jména Blåtand do angličtiny vzniklo slovo Bluetooth, které se nakonec uchytilo. Samotné logo technologie Bluetooth vzniklo z dvou vikingských run, které tvořily iniciály krále Haralda [13][14].

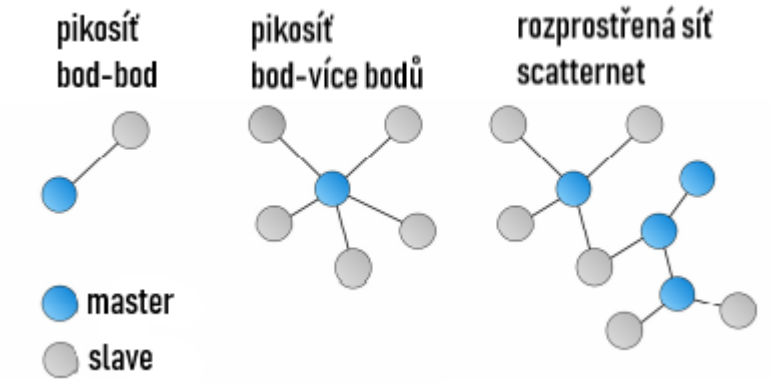


Obrázek 3: Vznik Bluetooth loga [14]

2.2 Topologie sítě

Komunikace přes Bluetooth je možná i mezi více než dvěma zařízeními mezi sebou. Funguje zde totiž spojení i více zařízení, spojených do tzv.**ad-hoc** sítě (bod-více bodů). V této sítí vystupuje jedno **nadřízené zařízení** (master), které řídí komunikační kanál a ke kterému se mohou kdykoliv snadno připojit nebo odpojit **podřízené zařízení** (slave). Rozlišujeme tři základní typy sítí:

- Pikosíť bod – bod (master – slave),
- Pikosíť bod – vice bodů (master – slaves),
- Rozprostřená síť



Obrázek 4: Topologie Bluetooth sítě [12]

V jedné pikosíti může být až sedm podřízených zařízení a další zařízení v tzv. suspended (pohotovostním) režimu (tzn. nespotřebovávají žádné síťové prostředky). Existuje zde také takt přeskakování – změna frekvence, který definuje master. Podřízené zařízení se s tímto taktem synchronizují – využití deterministické metody přístupu (*TDMA = Time Division Multiple Access*) [12].

Tyto podřízené jednotky mohou být zastoupeny i ve více sítích současně. Obdobně i master zařízení z jedné sítě může být podřízené v jiné síti. Takovéto překrývající se sítě tvoří rozprostřenou síť neboli scatternet [12].

2.3 Architektura

Bluetooth pracuje v bezlicenčním ISM pásmu (*Industrial Science & Medical*), které zahrnuje frekvence mezi 2,402 a 2480 GHz (podobně jako technologie Wi-Fi). Tento rozsah se dělí na 79 kanálů, kde každý má šířku 1 MHz [12] [13].

Takovéto Bluetooth zařízení využívá metodu přenosu v rozprostřeném spektru FHSS (*Frequency Hopping Spread Spectrum*), jehož princip spočívá v přeskakování mezi frekvencemi při přenosu za účelem co nejmenšího rušení. Počet přeskoků pro Bluetooth je 1600krát/s. Bluetooth zařízení jsou založené na vzájemné komunikaci pomocí **paketů** [12] [13].

2.4 Profily

Zařízení Bluetooth podporují také tzv. profily. Ty specifikují způsob, jakým může aplikační software přistupovat k různým vrstvám Bluetooth protokolové architektury [12].

Takovýchto profilů jsou pro Bluetooth řádově desítky. Pro tuto práci je nezbytná komunikace Bluetooth zařízení mezi sebou a také vzájemná výměna dat. Proto uvedení do této problematiky je potřeba znát následující dva profily:

2.4.1 GAP

Pomocí *Generic Access Profile* je BLE zařízením umožněná vzájemná komunikace. Dá se definovat jako nejvyšší kontrolní vrstva, která specifikuje, jak se mají zařízení chovat, aby umožnily vzájemnou výměnu dat. GAP profil definuje čtyři role zařízení:

- **Central** – Master zařízení, ke kterému může být v momentě připojeno více peripheral zařízení. Tyto zařízení inicializují spojení. Typicky jako central zařízení vystupují smartphony či tablety díky svému vysokému výpočetnímu výkonu.
- **Peripheral** – Slave zařízení. Jedná se o malá zařízení, vysílající advertising pakety, aby byly odhalitelné cental zařízeními. Vyznačují se, jak již bylo zmíněno, velmi nízkou spotřebou energie.
- **Broadcaster** – Broadcast je v GAP definován jako všesměrové vysílání. Typicky jako broadcaster vystupuje peripheral zařízení, které nedělá nic jiného, než že vysílá advertising pakety do svého okolí (beacon) a nepřijímá párování. Tyto pakety mají velmi omezenou velikost (max. 31 bytů).
- **Observer** – Jedná se o opačný případ broadcasteru. Úkolem observeru je pasivní skenování okolí a případné získávání a zpracování dat, která vysílá blízký broadcaster [11][12].

2.4.2 GATT

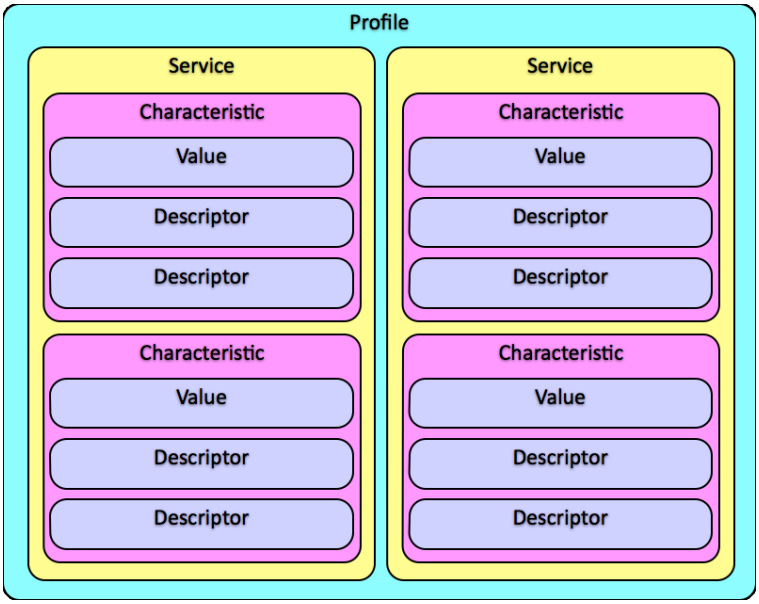
Generic Attribute Profile se zabývá postupy, jak mezi sebou dvě BLE zařízení posílají data za použití služeb a charakteristik. Jedná se o klíčovou část specifikace BLE, neboť pro práci s daty jsou jasně stanovená pravidla, podle kterých se musí každé BLE zařízení řídit.

Obdobně jako u GAP profilu i zde zařízení vystupují v rolích:

- **Server** – Jako GATT servery vystupují peripherals, která poskytují svá data.
- **Klient** – Za klienta většinou vystupují smartphony nebo tablety. Po navázání spojení mezi serverem a klientem si může klient vyžádat seznam služeb, které server nabízí.

Jako svůj transportní protokol pro výměnu dat mezi zařízeními využívá GATT Attribute Protocol (ATT). Data jsou hierarchicky organizována do skupin, nazývajících se **služby**, které v sobě obsahují podskupiny, zvané **charakteristiky** [11][12].

Každá charakteristika v sobě obsahuje data, spojená s danou službou. Charakteristika obsahuje vždy nejméně dva atributy, a to **characteristic declaration** (obsahující metadata) a **characteristic value**, což jsou samotná data. Mezi nepovinný atribut potom patří descriptor čili komentář, kterými se mohou rozšířit metadata. Charakteristiky mohou být definovány pro čtení (read) nebo pro zápis (write). Čtení charakteristiky provádí klient po zaslání tzv. read requestu (požadavku) a vrácená hodnota ze strany serveru, reagující na request, která je zapsaná jako characteristic value, jsou samotná data. Při zápisu se využívá write requestu, kde server následně vrací potvrzení, že hodnota byla úspěšně zapsána. Další vlastnost charakteristiky se nazývá write without response. Rozdíl oproti klasickému write requestu je zde to, že server potvrzení o zápisu dat klientovi nevrací. Využívají se ještě vlastnosti notify a indicate. Obě tyto vlastnosti zahajuje server. Vlastností notify si klient zajistí, že pokaždé, když se characteristic value změní, dostane patřičné upozornění. Na podobném principu je založena i indicate. Zde je rozdíl pouze v tom, že klient musí potvrdit přijetí upozornění [11].



Obrázek 5: Struktura GATT profilu [24]

Pro služby i charakteristiky platí, že jsou identifikovány pomocí tzv. **unikátních identifikátorů UUID** (*Universally Unique Identifier*). Ke dnešnímu dni existuje mnoho služeb, které mají svůj oficiálně schválený UUID podle Bluetooth SIG. Těmto službám je přidělena délka 16 bitů. Všechny ostatní služby a charakteristiky musí využívat UUID, dlouhé 128 bitů, které může být generované náhodně nebo vytvořené podle volby uživatele. V tabulce 1 níže jsou uvedeny rozsahy oficiálně schválených 16-bitových UUID [11].

UUID	00xx	18xx	27xx	28xx	29xx	2Axx
Popis	Namespace Descriptors	Services	Units	Declarations	Descriptors	Characteristics

Tabulka 1: *Oficiálně schválené 16-bitové UUID rozsahy [11]*

Shrnutím výše zmíněných částí GATT profilu by mohl být následující příklad služby např. pro Bluetooth Low Energy Lightbulb (žárovku):

Charakteristika	Stav žárovky	Nastavení tlumení světla	Spotřeba energie
UUID	FF11	FF12	FF16
Vlastnosti	Read, Write	Read, Write	Read
Hodnota	1	0x7F	340
Komentář	0 vyp, 1 zap	0x00 až 0xFF	Wh (watthodina)

Tabulka 2: *Ukázkový příklad služby* [11]

2.5 Vývoj standardu Bluetooth

Technologie Bluetooth prošla od svého počátku mnoha inovacemi, vylepšeními a optimalizacemi. V následujících řádcích je zmínka o každém stupni vývoje a jeho stručný popis:

2.5.1 První generace

První specifikace Bluetooth verze 1.0a a 1.0b byly provázeny problémy s kompatibilitou Bluetooth mezi zařízeními různých výrobců. Pozdější verze 1.1 a 1.2 již byly v praxi použitelné a byly oproti předchůdcům vylepšeny o nové funkce. To se psal rok 2002, respektive 2003. Maximální rychlost dosahovala v té době 1 Mbit/s a vzdálenost pro komunikaci byla do desíti metrů [12].

2.5.2 Druhá generace

Roku 2004 byla vydána specifikace 2.0. Přenosová rychlost se zvýšila až na 3 Mbit/s díky použití nové modulace. O tři roky později následovala verze 2.1, která jako první dostala podporu pro komunikaci blízkých polí NFC (*Near Field Communication*) – k párování dvou zařízení dochází zcela automaticky skrze těsné přiblížení. Tato generace také obsahovala systém bezpečného párování mezi zařízeními SSP (*Simple Security Pairing*) [12][13].

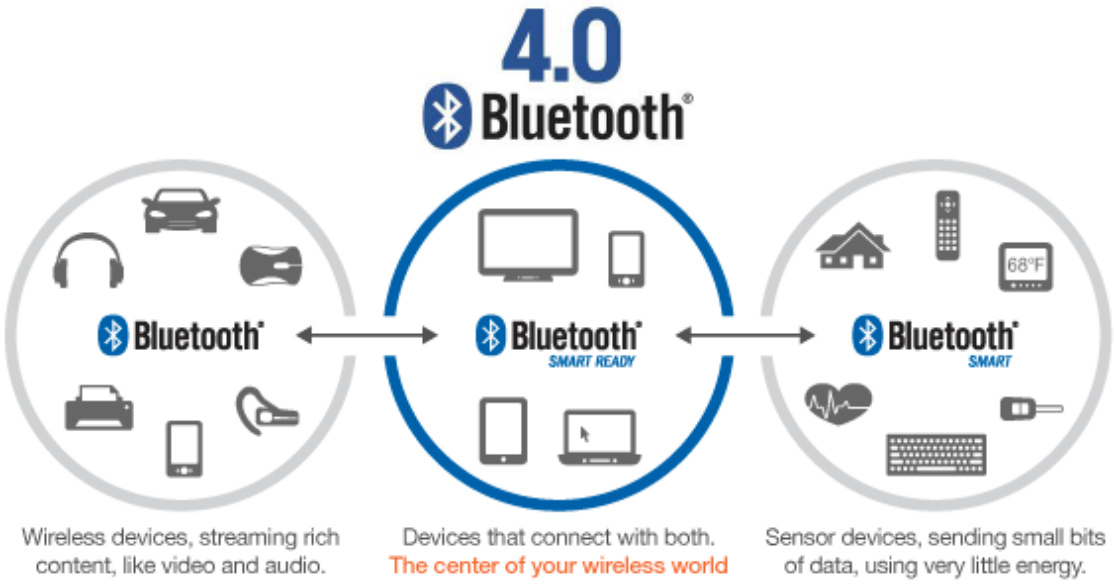
2.5.3 Třetí generace

Bluetooth verze 3.0 + HS (*High Speed*) byl vydán v roce 2009. HS v tomto případě zastává název pro podporu vysokorychlostního přenosu dat. Rychlost přenosu dat předchozí generace byla dostatečná pouze pro přenos dat menších objemů. S vývojem této specifikace však narostla maximální rychlost přenosu až na 24 Mbit/s, a tím umožnila přenos multimediálních dat o větších objemech, jako je například přenos video obsahu. Tato rychlost byla ovšem pro Bluetooth nereálná, a proto samotný vysokorychlostní přenos se provádí přes souběžné Wi-Fi spojení a samotný Bluetooth je zde implementován pouze k navázání spojení. Podpora vysokorychlostního přenosu dat nicméně není povinnou součástí specifikace a existuje také verze bez přípony HS. Mezi další vylepšení oproti předchůdcům patřilo např. vylepšení funkce kontroly napájení [13].

2.5.4 Čtvrtá generace

V roce 2010 přišla na svět i verze 4.0. Zahrnuje v sobě protokoly pro velmi nízkou spotřebu energie – Bluetooth LE (*Low Energy*). Díky tomu se zvýšila doba, po kterou jsou zařízení s verzí 4.0 schopná pracovat na baterii. velkou životnost baterie. Došlo také ke zvýšení maximálního dosahu, na kterou jsou zařízení schopna mezi sebou komunikovat, a to do cca 100 metrů. Hlavním cílem tohoto vylepšení bylo zaměřit se na požadavky moderní doby a implementovat Bluetooth zařízení do odvětví

jako jsou zdravotnictví, fitness, zabezpečovací technika, zábavní průmysl a další podobné oblasti, kde by se dalo takovéto zařízení využít. Když si tedy dnes koupíme např. nějaký ten nositelný elektronický náramek nebo nějaké chytré zařízení do domu, ve většině případů si můžeme být jisti, že využívá technologie Bluetooth LE. Další verze 4.1 a 4.2 přinášejí funkce, jako jsou například vylepšení bezpečnosti, zajištění menšího frekvenčního rušení jiných bezdrátových technologií jako je např. mobilní síť LTE (*Long Term Evolution*) nebo podpora IPv6 [12][13].



Obrázek 6: Základní rozdělení Bluetooth zařízení ve verzi 4.0 [15]

Jak lze z obrázku 6 výše vyčíst, verze 4.0 dělí Bluetooth zařízení na 3 základní třídy:

- **Bluetooth (classic)** - zařízení, přenášející velká množství dat; nedokáží komunikovat s třídou Smart,
- **Bluetooth Smart Ready** – nejvyšší třída; dokáže komunikovat se všemi ostatními třídami,
- **Bluetooth Smart** – přístroje senzorového typu s výrazně nižší spotřebou energie; komunikace pouze s typy Smart Ready [13] [15].

2.5.5 Pátá generace

Bluetooth 5.0, který byl vydán v roce 2016, je momentálně stále vyvíjen. Má přinášet výhody v podobě až čtyřnásobné zvýšení vzdálenosti, na kterou je možno komunikovat. Dvojnásobně zvýšená je také přenosová rychlost. Další novinkou je navýšení velikosti zpráv (z 31 bajtů na 255) u broadcastingu (bezdrátového vysílání), což se využije zejména v tzv. „beaconech“, rozšířená podpora pro Internet věcí (*IoT*), vyšší spolehlivost komunikace nebo větší orientace na bezpečnost [16].

3 Oblasti využití Arduino zařízení

Arduino má k dnešnímu datu ve své nabídce modelů mnoho desek, u kterých záleží jen na uživateli, podle jakých preferencí a potřeb si vybere požadovanou. Tyto desky prochází neustálým vylepšováním a dolaďováním. Na svých oficiálních stránkách dělí společnost Arduino své produkty na několik základních tříd:

- desky,
- moduly (menší varianty klasických desek),
- shieldy a nosiče (rozšiřující desky, dodávající projektu dodatečné funkce),
- kity a příslušenství.

Dále jsou zde produkty rozděleny do skupin podle své funkce, závislé na zkušenosti uživatele. Tyto skupiny by se daly nazvat následovně:

- Vstupní úroveň – jednoduché použití, ideální pro základní projekty, jsou zde nejvhodnější produkty k úspěšnému naučení se základům programování s Arduinem.
- Rozšířené funkce – k sestrojení složitějších projektů, pokročilé funkce, popř. větší výkon.
- Internet of Things – zařízení, umožňující vyměňovat si data a pracovat online.
- Vzdělání – skupina vhodná pro školství, kde se mohou studenti inspirovat ve světě programování ve spojení s elektronikou.
- Wearables – nositelná chytrá elektronika, využívaná např. ve fitness odvětví [6].

3.1 Arduino klony

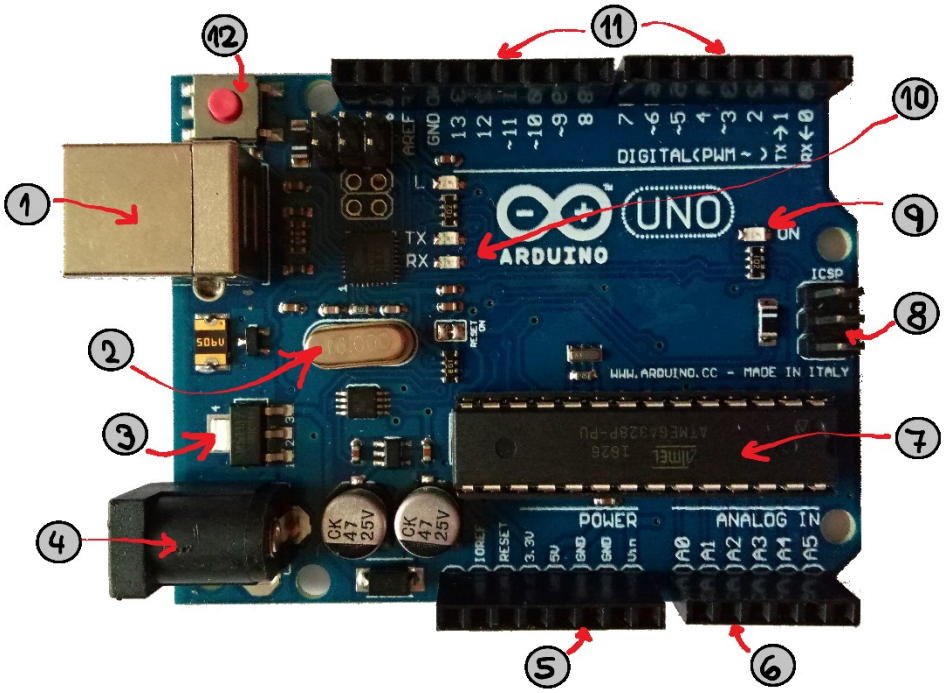
Vzhledem k tomu, že platforma Arduino je open-source a dokumentace je volně dostupná online, vyrábí tyto produkty i další výrobci. Arduino klon je většinou pojmenován koncovkou "duino". Celé slovo Arduino je totiž chráněno licencí. Tyto klony se od originálního Arduina v podstatě neliší. Často obsahují pouze konkrétní vylepšení (např. více konektorů, spínaný zdroj napájení apod.). Samozřejmostí je opět dobrá dostupnost návodů i pro tyto klony. Některé klony se od originálu liší pouze barvou, ovšem jiné mohou být podobné i méně, a to např. kvůli jiného tvaru desky či jiným použitým hardwarem.

Klony jsou založeny většinou na stejných procesorech jako originály, tedy řadě ATmega, ovšem můžeme narazit i na čip ESP8266 nebo na použití jiných převodníků atp. V takových případech mohou být problémy se samotným zprovozněním desky, a ne vždy je lehké najít jejich příčinu.

Pokud uživatel váhá, zda je lepší si koupit levnější klon, u kterého ovšem není stoprocentně zajištěna srovnatelná kvalita s originálem, to už záleží pouze na jeho preferencích [5][7].

3.2 Volba Arduino produktu

Pro tuto práci, kde je potřeba pomocí Arduino desky připojit Bluetooth modul, byla zvolena deska Arduino UNO R3.Tato deska je v následující části práce blíže specifikována.



Obrázek 7: Arduino UNO R3

1	USB konektor	Slouží k propojení desky s počítačem pomocí USB kabelu.
2	Krystalový oscilátor	Pomocí oscilátoru se deska vypořádává s časovými operacemi. Číslo, udávané na horní straně, značí frekvenci v kHz.
3	Regulátor napětí	Zajišťuje regulaci napětí, kterou deska dostává a stabilizuje napětí na ostatních součástkách.
4	Napájecí konektor	Deska může být napájena také připojením stejnosměrného napájecího zdroje (6-20V)
5	Napájecí piny	Tyto piny slouží k napájení desky 3,3V nebo 5V napětím, dále je zde pin na uzemnění, připojení externího napájecího zdroje nebo pin na resetování desky
6	Analogové piny	Převádí analogové signály např. ze senzorů na digitální hodnoty, se kterými následně pracuje mikroprocesor
7	Mikroprocesor	Je jakýmsi "mozkem" desky. Informace o mikroprocesoru jsou napsány na jeho vrchní straně.
8	ICSP piny	Využity, pokud je Arduino bootloader (program, používaný k programování desky) poškozen nebo chybí.
9	LED indikátor napájení	Tato LED dioda začne svítit, pokud úspěšně připojíme naši desku ke zdroji napájení.
10	Tx a Rx LED diody	Jsou zodpovědné za sériovou komunikaci. Tx (transmit) bliká, pokud se posílají sériová data a Rx (receive) bliká během procesu přijímání dat. (spojeny s digitální piny 0 a 1)
11	Digitální I/O piny	Mohou být nakonfigurovány jako digitální vstupy pro čtení logických hodnot nebo jako výstupy pro ovládání ostatních modulů, shieldů atd. Piny, označené "~" mohou generovat PWM (pulzně šířkovou modulaci).
12	Reset tlačítko	Slouží k resetování naší Arduino desky, tzn. Náš kód se spustí znovu od začátku.

Tabulka 3: *Popis desky Arduino UNO R3* [8][9][10]

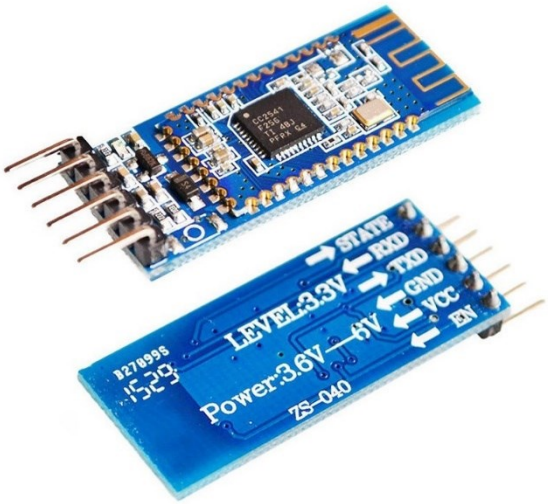
Pro zpracování bakalářské práce byla zvolena právě tato deska, neboť obsahuje všechny piny, potřebné k propojení s Bluetooth modulem vybraným níže, a také kvůli její dobré ceně a dostupnosti. Obecně je jednou z nejpoužívanějších, ne-li nejpoužívanější Arduino deskou vůbec.

3.2.1 Specifikace zvoleného Arduino zařízení

Vývojová deska Arduino UNO R3 využívá standardně čip ATmega328P, navržený firmou Atmel. Pracuje s napětím 5V, přičemž doporučené hodnoty vstupního napájení jsou v rozmezí 7-12V. Každý digitální I/O pin dovoluje odběr 20 mA, kdežto 3.3V pin dovoluje až 50 mA. Co se týče pamětí, jejich velikosti vycházejí z čipu ATmega328P. Flash paměť neboli místo, vyhrazené pro Arduino sketch, zabírá 32 kB. SRAM (v překladu *Static Random Access Memory*) je zase místo, kde sketch vytváří proměnné a manipuluje s nimi. Velikost této paměti je u tohoto zařízení 2 kB. Jako poslední typ paměti je podporován i typ EEPROM (*Electrically Erasable Programmable Read-Only Memory*). Jedná se o jednoduchou paměť o velikosti 1 kB, sloužící k permanentnímu uložení informací. Zůstanou zde tedy i po odpojení napájení [11].

3.3 Volba Bluetooth modulu

Aby bylo splněno zadání této bakalářské práce, musel být zvolen Bluetooth takový modul, který podporuje verzi 4.0 a musí být schopen komunikovat s deskou Arduino UNO R3. Vybrat si produkt, splňující tyto kritéria, nebyl problém, neboť na internetových obchodech je takovýchto kompatibilních modulů více. Byl vybrán modul Bluetooth 4.0 BLE HM-10.



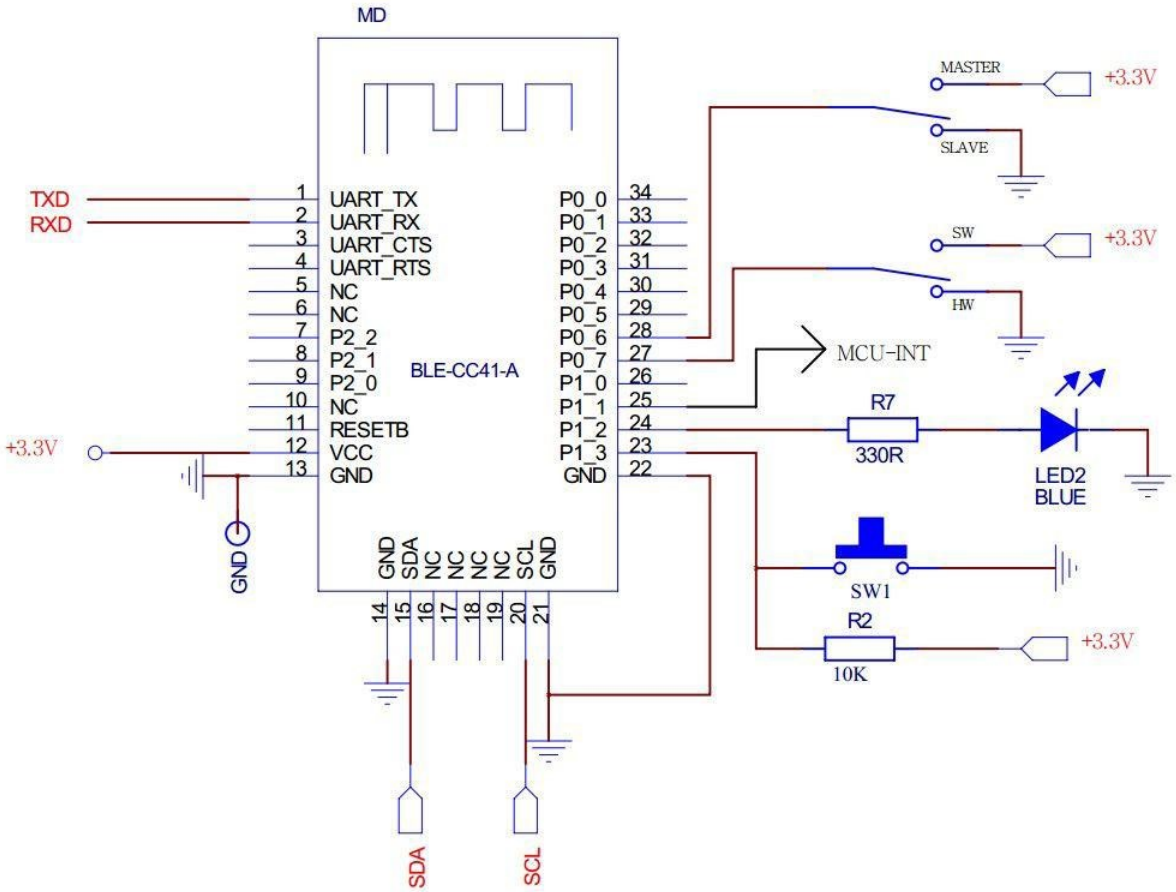
Obrázek 8: Modul Bluetooth 4.0 BLE HM-10 [23]

3.3.1 Specifikace zvoleného BLE modulu

Komunikační Bluetooth modul HM-10 vysílá na frekvenci 2.4 GHz v bezlicenčním ISM pásmu. Je osazován čipem CC2540 (popř. CC2541). Udávaná hodnota napájení je pro tento modul stanovena na +3,3V a maximální povolený proudový odběr modulu je 50mA. V aktivním stavu odebírá tento modul 8.5mA. S Arduino zařízením probíhá vzájemná komunikace pomocí sériové linky, kde výchozí rychlost je 9600 baudů. Maximální nastavitelná rychlost je u tohoto modulu až 230400 baudů. Pro tuto komunikaci je nutno mít propojen Bluetooth modul s Arduinem pomocí čtyř vodičů – zdroj napětí, uzemnění, vysílání a příjem informací – viz v následující kapitole. Samotná konfigurace modulu je

podrobněji specifikována v dalších kapitolách této práce a to pomocí AT příkazů, které se píší do sériového monitoru v programu Arduino IDE. Tyto příkazy jsou dohledatelné buď v technické dokumentaci produktu či pomocí samotného příkazu „AT+HELP“.

Na obrázku 9 níže je zobrazeno schéma modulu HM-10. Z něj je zřejmé, že modul obsahuje kromě šesti pinů, vyvedených pomocí kolíků (viz obrázek 8) i další piny. Ty však nejsou pro tuto práci nutné využívat [17].



Obrázek 9: Schéma Bluetooth modulu HM-10 [25]

4 Návrh řešení praktické části práce

Samotným cílem praktické části práce bylo zajištění bezdrátové komunikace pomocí technologie Bluetooth 4.0 mezi mobilním zařízením s operačním systémem Android a zařízením Arduino. V průběhu řešení práce nebylo dostupné vývojové prostředí pro vývoj aplikací s operačním systémem iOS. Z tohoto důvodu byla vyvinuta verze jen pro zařízení s operačním systémem Android. V případě operačního systému iOS by byla funkcionalita aplikace stejná a aplikace pro iOS zařízení může být vyvinuta v případné navazující práci.

Tato komunikace je zajištěna pomocí Bluetooth GATT profilu (*Generic Attribute Profile*), který, jak již bylo zmíněno dříve, definuje, jak si mezi sebou BLE zařízení posílají data za použití služeb a charakteristik. Praktická část práce byla vyvíjena ve dvou vývojových prostředích. Pro aplikaci, která je nainstalována na mobilním zařízení s operačním systémem Android, bylo využito prostředí Android Studio 3.3.1 a pro druhou část, zajišťující správu zařízení Arduino UNO R3 bylo použito prostředí Arduino IDE 1.8.9.

4.1 Možnost využití v praxi

Příkladem využití této bezdrátové komunikace by mohla být situace na různých veletrzích či konferencích. Každý člověk, který by tyto akce navštívil (a měl u sebe mobilní zařízení s operačním systémem Android), by měl možnost dostat u vstupu zařízení, vyvinuté při řešení této bakalářské práce, které by umožňovalo automaticky získávat vizitky lidí, kteří jsou v dosahu jeho Bluetooth skeneru a sami svou vizitku zrovna vysílají. Tímto shromážděním dat by měl uživatel následně přehled o lidech, kteří se vyskytovali na akci v jeho blízkosti.

Obrázek 10: Využití komunikačního zařízení v praxi [28]

Vezměme si ku příkladu člověka, který jde na veletrh práce, kde lidé nabízejí zajímavá pracovní místa a hledají nové kolegy či kolegyně. Tento uživatel (vybavený mobilním zařízením Android) by dostal při vstupu do areálu na vypůjčení mnou vytvořené zařízení a následně by byl požádán o stáhnutí a nainstalování mé vytvořené aplikace do svého mobilního zařízení. Do této aplikace by návštěvník zadal své osobní údaje (jméno, příjmení, věk a své záliby) a jednoduchým připojením se k Bluetooth modulu konkrétního Arduino zařízení a následovaným stiskem tlačítka pro nahrání dat by nahrál do tohoto zařízení svou vizitku, aby mohli lidé získat i jeho data a vědět, o co se uživatel zajímá nebo co patří mezi jeho koníčky. Poslední krok, který by musel provést pro start skenování a vysílání, by bylo jednoduché zadání časového úseku, na jak dlouhou dobu by chtěl vizitky z okolí získávat a zároveň sdílet svou vizitku s ostatními. Během této doby by tento uživatel navštěvoval stánky/konferenční místnosti lidí, kteří by zrovna například prezentovali své firmy nebo společnosti široké veřejnosti a byli tím pádem momentálně časově nedostupní v komunikaci s jedinci. Díky komunikačnímu zařízení by se však k uživateli automaticky dostala všechna data, která by prezentující měl ve své vizitce, aniž by došlo k vzájemné verbální komunikaci.

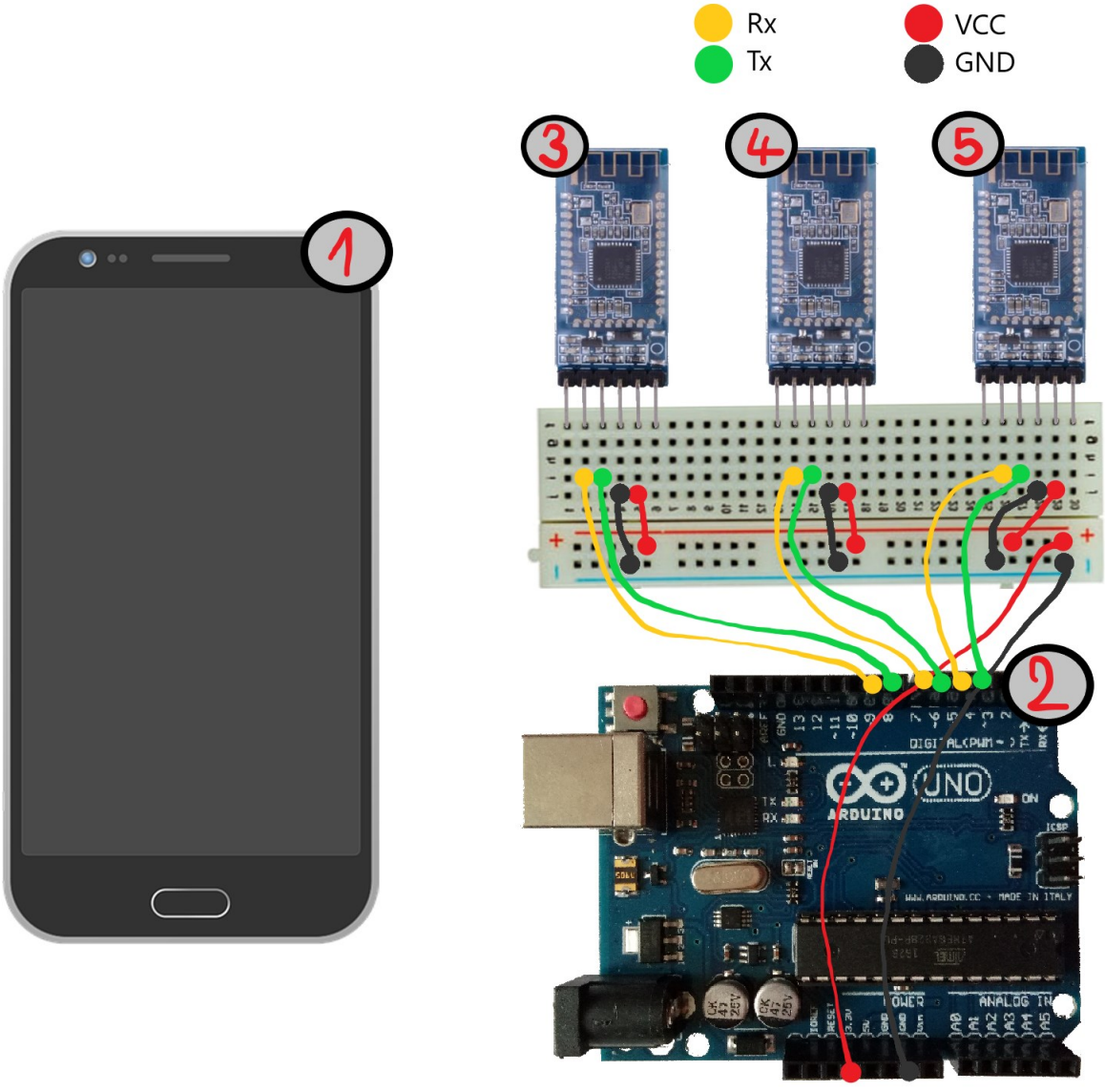
Tímto stylem by si potom uživatel v Android aplikaci zobrazil seznam získaných vizitek, z nichž by získal informace a postřehy, které během prezentace vůbec nezazněly nebo kdy je nezachytil, případně si je nezapamatoval. Mezi zájmy prezentujícího/firmy by také mohly být udány informace, podle kterých by se uživatel rozhodoval, koho nakonec osloví a kdo mu bude podle jeho zájmů a celkového profilu bližší.

Tento příklad je pouze jednou z možností, kde by se dalo zařízení využít. Našly by se určitě i jiné situace, kde by se mohlo toto vytvořené komunikační zařízení v kombinaci s Android aplikací použít.

4.2 Návrh schématu pro propojení Arduino zařízení s BLE moduly

Praktická část práce, ve které má uživatel možnost shromažďovat data (v našem případě vizitky) ostatních uživatelů a zároveň s nimi svá data sdílet, byla vypracována dle schématu, vyskytujícím se na obrázku 11. Vypozorovat z něj lze následující části:

- mobilní zařízení s operačním systémem Android (1),
- Arduino UNO R3 (2),
- tři Bluetooth Low Energy HM-10 moduly, vystupujících v rolích Beacon, Scanner a Receiver (3, 4, 5)
- nepájivé kontaktní pole a propojovací vodiče



Obrázek 11: Schéma propojení Arduino zařízení s BLE moduly

Pro propojení každého HM-10 modulu s vývojovou deskou Arduino je potřeba čtyř propojovacích vodičů, přičemž tyto 4 vodiče se mohou rozdělit dle funkce na dvě dvojice. První dvojice zajišťuje napájení, kde červená barva vodiče znamená přísun napětí +3.3V a černá uzemnění (GND).

Druhá dvojice slouží pro transport dat mezi modulem a Arduino deskou, kde žlutá barva zajišťuje příjem dat (receive) a zelená vysílání dat (transmit). Další dva vodiče jsou zde nutné pro zavedení napětí a uzemnění mezi samotnou Arduino deskou a nepájivým kontaktním polem, které je zde využito pro kompaktnost výsledného propojení. Tyto dva vodiče tvoří dva kanály (napětí a uzemnění), ze kterých jsou následně vyvedeny vodiče pro každý Bluetooth modul zvlášť – viz obr. 10 výše.

V následujících podkapitolách je každá část schématu popsána zvlášť, aby bylo zřejmé, k čemu přesně slouží a jaká je její úloha ve výsledném komunikačním zařízení.

4.2.1 Mobilní zařízení s OS Android

Jak již bylo zmíněno v úvodu kapitoly 4, pro toto zařízení byla vyvinuta aplikace ve vývojovém prostředí Android Studio 3.3.1. Toto prostředí využívá programovacího jazyka Java, na kterém je založeno samotné uživatelské rozhraní operačního systému Android. Pomocí této aplikace řídí uživatel veškerou komunikaci mezi mobilním zařízením a Bluetooth moduly, připojenými se zařízením Arduino.

4.2.2 Arduino UNO R3

Kromě toho, že slouží pro propojení všech Bluetooth modulů, poskytuje také permanentní úložiště vizitky uživatele skrze paměť EEPROM. Na toto zařízení je nutné nahrát sketch (zdrojový kód), vyvinutý ve vývojovém prostředí Arduino IDE 1.8.9, pomocí kterého má Arduino možnost využít Bluetooth moduly a komunikovat s nimi. Detailní popis prostředí Arduino IDE je popsán v kapitole 1.4. K napájení samotné desky je zde využito rozhraní USB, připojené k počítači, přes který se daný sketch na tuto vývojovou desku nahraje. Další možností napájení by mohl být zdroj stejnosměrného napětí v rozmezí 9 až 12V, nicméně pro tuto práci je volba USB rozhraní ideální, a to z důvodu využití sériového monitoru vývojového prostředí Arduino IDE, který umožňuje sledovat výstupy zdrojového kódu za běhu programu.

4.2.3 HM-10 BLE moduly

Součástí výsledného komunikačního zařízení jsou také tři Bluetooth moduly HM-10. Každý z nich má svou speciální funkci, která je odlišná od ostatních.

Jednu z funkcí zastupuje modul, který byl pojmenován **Receiver**. Právě přes tento modul se uživatel musí připojit pomocí svého mobilního zařízení a tím tvořit dvojici central (mobilní zařízení) – peripheral (BLE modul) dle GAP profilu a zároveň klient – server dle profilu GATT. K danému modulu (serveru) může být v určitý okamžik připojeno pouze jedno mobilní zařízení (klient), jinak se modul stává neviditelným pro ostatní mobilní zařízení do doby, než se právě spojené mobilní zařízení od modulu odpojí. Po připojení mobilního zařízení k tomuto modulu má uživatel možnost řídit skrze aplikaci, vytvořenou v Android Studiu, veškerou komunikaci, která bude probíhat s vývojovou deskou Arduino.

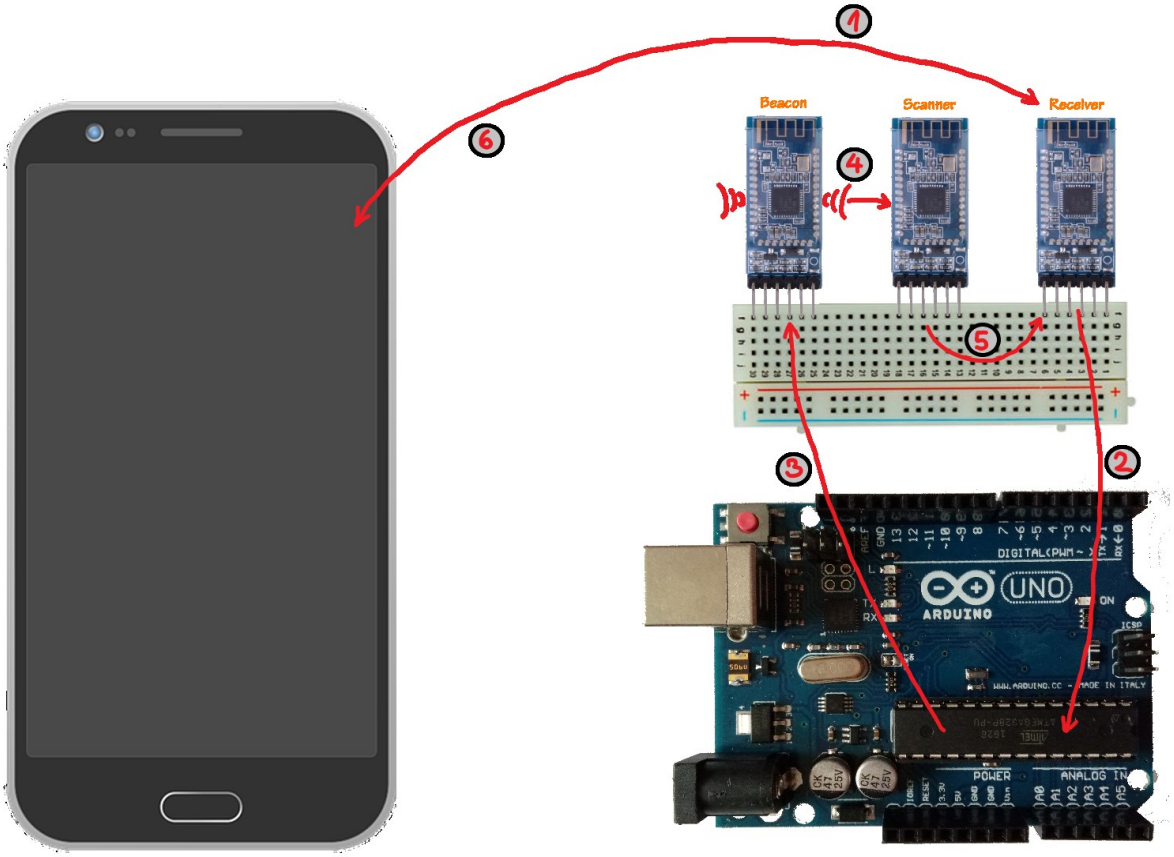
Účelem modulu, pojmenovaném **Beacon**, je vysílání vizitky uživatele do svého okolí, které je závislé na dosahu signálu. Dle GAP profilu je tento modul v roli broadcasteru. K němu se tedy žádné mobilní zařízení nepřipojuje a slouží pouze jako vysílač/beacon, který vysílá data do svého okolí.

Jako poslední část výsledného komunikačního zařízení má svou funkci zastoupen i modul, jehož název byl nastaven na **Scanner**. Dle GAP profilu se jedná o pravý opak broadcasteru – tedy observer.

Tento modul má tím pádem na starost získávání dat z BLE zařízení, která mají tento modul v dosahu svého signálu. Měl by tedy zachytit i mou vizitku, která vyšle svá data skrze vysílač (beacon).

4.3 Průběh komunikace

Na obrázku 12 je graficky vyobrazen průběh komunikace. Následuje popis každé části, kterou znázorňuje číslice.



Obrázek 12: Průběh komunikace mezi jednotlivými zařízeními

1. První část komunikace musí proběhnout mezi mobilním zařízením a modulem HM-10. Uživatel se přes aplikaci připojí k receiver modulu. Toto připojení (mobilní zařízení – receiver) musí trvat během celého průběhu veškeré komunikace – tedy během všech šesti částí dle obrázku 12. Po úspěšném připojení je uživatel schopen komunikovat s Arduino vývojovou deskou, neboť zdrojové kódy aplikace a zároveň Arduino vývojové desky jsou nastaveny na naslouchání změn hodnoty charakteristiky (characteristic value) určité stejné služby, která se nachází v BLE modulu - viz kapitola 2.4.2. Díky tomuto vzájemnému naslouchání je následně možné odeslat data vizitky, kterou si vyplníme v aplikaci. Po stisknutí tlačítka „Nahrát vizitku do Arduina“, které je součástí aplikace, proběhne sériový přenos dat po jednom bytu z mobilního zařízení do BLE modulu.
2. Tento bod komunikace přímo navazuje na bod 1. Data, která receiver přijal, jsou ihned zpracována a uložena do EEPROM paměti Arduino zařízení, kde jejich uložení zůstává natrvalo (pokud je později nepřepíšeme).

3. Až se uživatel rozhodne, že chce začít svou vizitku vysílat a skenovat data (vizitky) z okolí, stiskne příslušné tlačítko v aplikaci, pomocí kterého vyšle mobilní zařízení receiveru znak – tzn. jeden byte. Zdrojový kód Arduina tento znak zpracuje a podle něj se spustí část kódu, která získá data z paměti EEPROM a upraví je do takové formy, aby se daly vysílat v advertising paketech přes beacon modul.
4. V tomto bodě jsou data vizitky rozdělena do paketů (viz kapitola 6.2.2). Zdrojový kód vývojového prostředí Arduino IDE zajišťuje vysílání těchto paketů periodicky dle nastavené délky vysílání jednoho paketu. Ta byla stanovena na 4 sekundy s ohledem na rychlost sériové komunikace modulu pro skenování. Během těchto čtyř sekund se vždy vysílá paket s určitou částí dat vizitky a také zároveň probíhá skenování zařízení z okolí. Celková délka vysílání a skenování záleží také na tom, jak dlouhý interval v sekundách uživatel v aplikaci zadal. O této problematice však pojednává kapitola 6.2.3.
5. Z důvodu omezené kapacity úložiště pro proměnné na straně Arduino zařízení se data získaná skenováním ihned po každém vyslaném paketu nastavují jako hodnota charakteristiky do receiver modulu. Během tvorby zdrojového kódu bylo zjištěno, že pokud by se všechna data, získaná během skenování ukládala do proměnných a vyslala se ve větším objemu najednou, nezbylo by již žádné další místo v paměti pro následný bezproblémový chod celého programu.
6. Na stejném principu zmíněném v bodu 1 – tedy skrze hodnoty charakteristiky receiver modulu – se veškerá naskenovaná data získaná během skenování postupně odesílají do mobilního zařízení. Tam probíhá zpracování veškerých dat, vzniklých skenováním a následně doručených přes receiver modul. Ta následně procházejí filtrem zjišťujícím zda při jejich přenosu nedošlo ke znehodnocení. Pokud jsou data určité vizitky v pořádku, vizitka se připíše (popř. upraví stávající vizitku) do/v seznamu vizitek v aplikaci.

5 Otestování dosahu a náročností na elektrickou energii

Maximální vzdálenost, na kterou mohou veškerá bezdrátová zařízení komunikovat, obecně závisí na mnoha faktorech, jako je např. prostředí, design antény, orientace či kryt zařízení. Dosah a kvalita spojení u Bluetooth Low Energy zařízení také záleží na tom, jestli jsou mezi komunikujícími zařízeními překážky. Vysílací výkon, měřený v dBm (decibelech vztažených na miliwatty), se u BLE zařízení obvykle pohybuje v hodnotách mezi cca -30 dBm až 0 dBm. Platí, že čím blíže je uživatel k danému vysílajícímu zařízení, tím vyšší hodnoty dBm je schopen naměřit. Také platí to, že čím vyšší je vysílací výkon, tím vyšší jsou požadavky na elektrickou energii daného zařízení, což ovlivňuje celkovou životnost článků baterie.

Co se týče zařízení Arduino UNO R3 (jak již bylo zmíněno v kapitole 3.2.1), to pracuje s napětím 5V, přičemž doporučené hodnoty vstupního napájení jsou v rozmezí 7-12V. Pin 3,3V, zajišťující napětí pro celé nepájivé kontaktní pole, na kterém jsou všechny BLE moduly, dovoluje maximální proudový odběr 50 mA. Vzhledem k tomu, že HM-10 moduly jsou na kontaktním poli celkově tři a každý odebírá v maximálně 8,5 mA, je 3,3V pin s maximálním proudovým odběrem 50 mA dostačující [11].

Bluetooth Low Energy moduly HM-10, použité v této práci, byly nastaveny na vysílací výkon 0 dBm. Tento údaj se dá u každého modulu zvlášť zjistit s pomocí vývojového prostředí Arduino IDE a nahrání příslušného sketche, který umožňuje komunikaci mezi uživatelem a modulem pomocí AT příkazů. Konkrétní AT příkaz pro zjištění vysílacího výkonu je „AT+POWE?“, kde odpovědí na tento příkaz nám bude parametr (číslo), podle kterého si v technické dokumentaci modulu zjistíme odpovídající hodnotu výkonu [17].

V rámci maximální vzdálenosti, ze které je uživatel schopen vytvořené komunikační zařízení zachytit, bylo provedeno praktické měření za pomoci několika mobilních zařízení, vybavených Bluetooth anténami. Toto měření probíhalo v otevřeném prostředí a zařízení se nacházela v přímé viditelnosti mezi sebou. Bylo zjištěno, že vytvořené komunikační zařízení je v použitých mobilních zařízeních viditelné (dosažitelné) až do vzdálenosti zhruba 25 metrů. Typická vzdálenost, na kterou mezi sebou Bluetooth zařízení komunikují, se uvádí v jednotkách metrů, nicméně naměřená hodnota 25 metrů je stále dosažitelná a reálná hodnota. Zjištěný vysílací výkon se při tomto měření pohyboval kolem -93 dBm. Se snižováním vzájemné vzdálenosti se měřený výkon zvyšoval a při téměř nulové vzdálenosti zařízení vzrostl na cca -28 dBm, což odpovídá zhruba 0,0016 mW. Pro převod hodnoty z dBm na mW lze využít vzorec 1 uvedený níže [18].

$$P_{dBm} = 10 \cdot \log 10 \left(\frac{P_{mW}}{1\,mW} \right) \quad (1)$$

Úpravou lze získat vzorec 2, do kterého již stačí jen dosadit příslušný výkon v dBm, který je v našem případě -28 dBm [18].

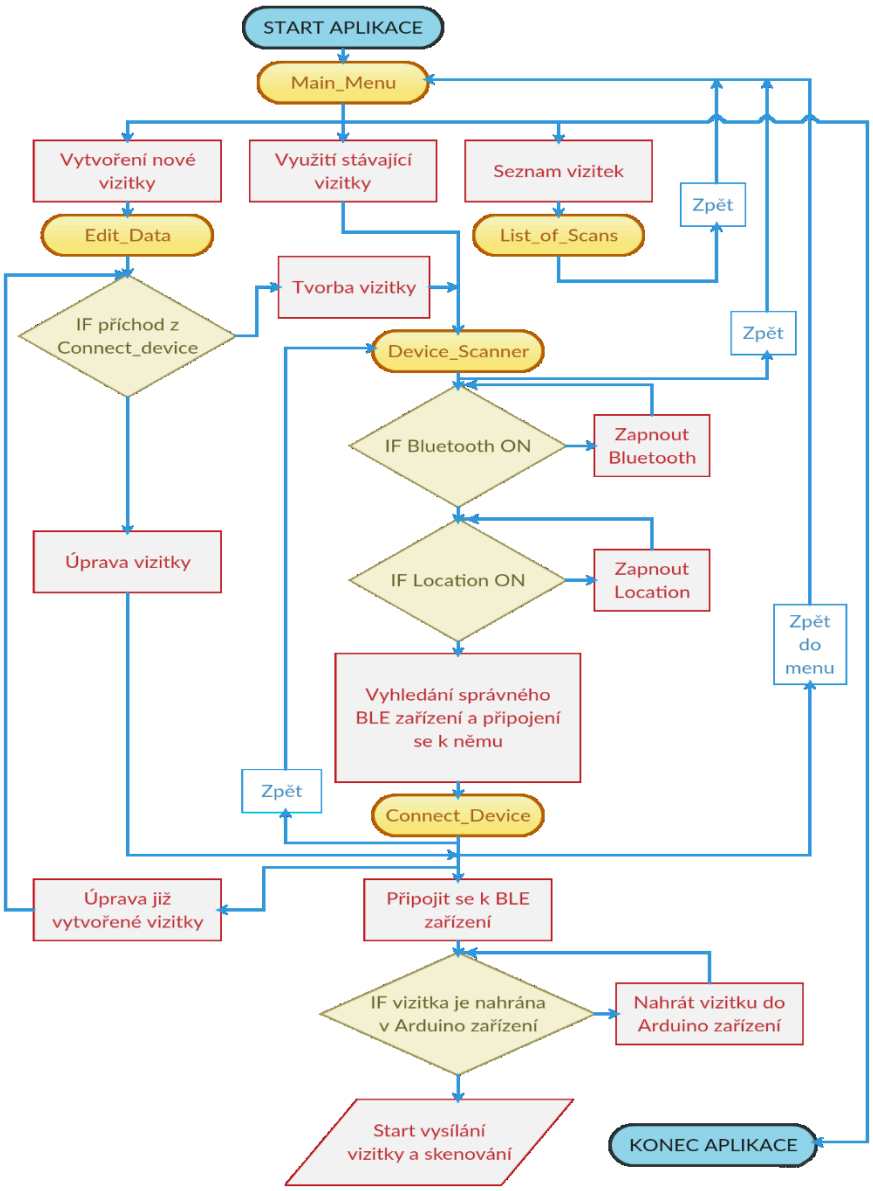
$$P_{mW} = 1\,mW \cdot 10^{\left(\frac{P_{dBm}}{10} \right)} \quad (2)$$

6 Zpracování dokumentace navrženého řešení

Tato kapitola pojednává o vývoji aplikace navržené v programu Android Studio 3.3.1 a o popisu důležitých částí jejího zdrojového kódu. Dále je zde také zdokumentován zdrojový kód, nahrávaný na Arduino zařízení.

6.1 Vývoj aplikace pro Android zařízení

Název aplikace je Bluetooth Beacon. Skládá se z pěti aktivit. Aktivita je stěžejní součástí samotné Android aplikace. Na rozdíl od programových paradigmat, ve kterých se aplikace spouští metodou main(), systém Android iniciuje kód skrze aktivity voláním specifických callback metod, které odpovídají částem životního cyklu samotné aktivity. Každá z pěti aktivit, využitých pro chod aplikace, má tedy pro celkovou funkčnost této aplikace svůj účel. Obrázek 13 znázorňuje vývojový diagram celé vytvořené Android aplikace [27].



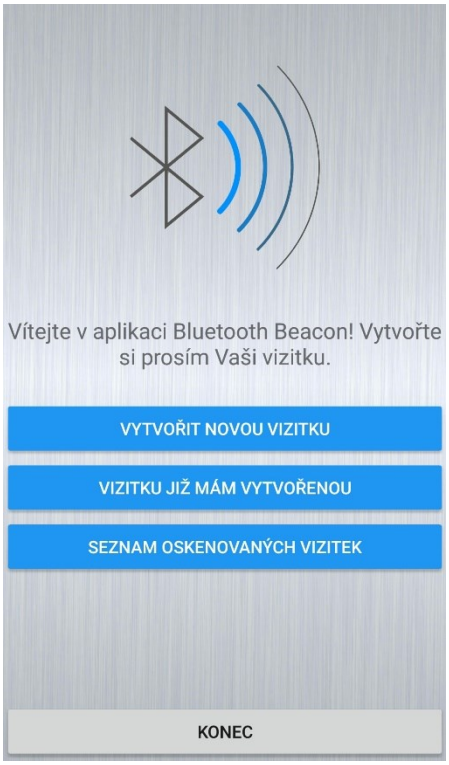
Obrázek 13: Vývojový diagram Android aplikace

6.1.1 Hlavní menu aplikace

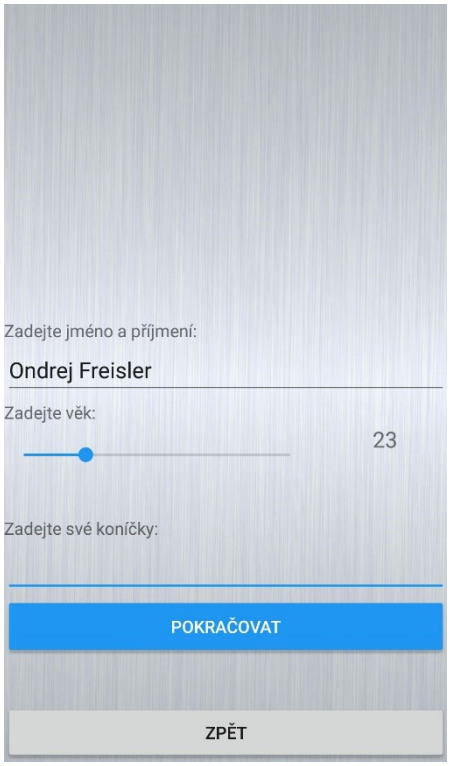
Úvodní aktivitou celé aplikace je hlavní menu (aktivita Main_Menu – obr. 13), ve kterém má uživatel možnost výběru další akce pomocí tlačítek. Může si vytvořit novou vizitku, která ho odkáže do aktivity pro vytvoření nové vizitky nebo přeskočit tento krok, pokud si vizitku vytvořil již dříve. Další možností je zobrazení seznamu vizitek a poslední tlačítko slouží k ukončení celé aplikace.

6.1.2 Vytvoření/úprava dat

Při kliknutí na vytvoření nové vizitky se uživateli otevře aktivita Edit_Data (obr. 14), ve které vyplní do příslušných polí jednotlivá data. Nesmí se však využívat znaky, obsahující diakritiku, neboť s daty se později pracuje jako s decimálními (popř. hexadecimálními) hodnotami z ASCII tabulky (*American Standard Code for Information Interchange*). Je zde také nastavena minimální a maximální délka každé položky vizitky. Uložení dat se potvrdí stiskem tlačítka „Pokračovat“. Tato data jsou následně vepsána do souboru, který se automaticky vytvoří v úložišti mobilního zařízení (popř. přepíše).



Obrázek 14: Aktivita Main_Menu



Obrázek 15: Aktivita Edit_Data

6.1.3 Skenování zařízení a jejich zobrazení v seznamu

Jako další se dostává na řadu aktivita Device_Scanner. Jejím úkolem je vyhledání námi požadovaného BLE modulu a jeho následné vybrání. Pro úspěšný start skenování BLE zařízení je však nutností mít na mobilním zařízení zapnutý Bluetooth společně s přístupem k poloze. Pokud jsou tyto dvě podmínky splněny, je aplikace schopna spustit sken dostupných BLE zařízení v okolí. Se samotným spuštěním skenování, které je nastaveno na 10 sekund, se spustí také instance třídy AsyncTask. Úkolem této instance je řízení procesu (v tomto případě řízení komponenty ProgressBar) na pozadí aplikace – tzv. background process, a následné zajištění callback funkce po jejím ukončení. Poté, co tedy uživatel

klikne na spuštění skenování zařízení, se ve zdrojovém kódu aplikace společně s ostatními příkazy volá funkce startScan(BLEScanCallback). Výstupem této funkce je přidání veškerých zjištěných BLE zařízení v okolí do seznamu. Pro získání dat o jednotlivých BLE zařízeních a soudružnosti těchto dat pospolu byla vytvořena třída DeviceHolder. Její implementace je následující:

```
class DeviceHolder{
    private BluetoothDevice device;
    private int rssi;

    DeviceHolder(BluetoothDevice device, int rssi){
        this.device = device;
        this.rssi = rssi;
    }

    BluetoothDevice getDevice() return device;
    String getAddress()         return device.getAddress();
    String getName()            return device.getName();
    void setRSSI(int rssi)      this.rssi = rssi;
    int getRSSI()               return rssi;

}
```

Pro zobrazení veškerých naskenovaných zařízení ve společném seznamu byla vytvořena třída DeviceList, která je vlastním rozšířením předdefinované Android třídy ArrayAdapter. Jednotlivými položkami této třídy jsou právě instance třídy DeviceHolder. Za zmínku dále stojí znázornění funkce startScan a její parametr BLEScanCallback. Tento parametr zajišťuje přidání jednotlivých instancí třídy DeviceHolder do společného seznamu (Android komponenta ListView).

```
private DeviceList adapter;
private BluetoothLeScanner BTScanner;
private BluetoothAdapter BTAdapter;
public BluetoothManager BTManager;
ArrayList<DeviceHolder> devicesHolder;

BTManager = (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
LOCManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
BTAdapter = BTManager.getAdapter();

public void startScanning() {
    BTScanner = BTAdapter.getBluetoothLeScanner();
    adapter = new DeviceList(devicesHolder, getApplicationContext());
    listView.setAdapter(adapter);
    BTScanner.startScan(BLEScanCallback);
}

private ScanCallback BLEScanCallback = new ScanCallback() {
    @Override
    public void onScanResult(int callbackType, ScanResult result) {
        BluetoothDevice device = result.getDevice();
        int rssi = result.getRssi();
        DeviceHolder deviceHolder = new DeviceHolder(device, rssi);
        adapter.addDevice(deviceHolder, rssi);
        adapter.notifyDataSetChanged();
    }
};
```

6.1.4 Zajištění komunikace s BLE moduly

Po výběru příslušného BLE zařízení (v našem případě receiveru) se dostává na řadu další aktivita jménem Connect_Device. Té je věnována největší část zdrojového kódu aplikace. Pro funkčnost této aktivity je nutné mít na Arduino zařízení nahrán sketch, neboť mezi BLE moduly, připojeny k Arduino zařízení, a touto aktivitou již probíhá přímá komunikace. Prvním bodem, který je pro následnou funkčnost zásadní, je připojení se k vybranému HM-10 modulu a následnému získání dostupných služeb a charakteristik, které tento modul obsahuje. Cílem tohoto procesu je získat přístup k proměnné, přes kterou BLE modul komunikuje (nastavuje characteristic value). Tento proces shrnuje pseudokód [26]:

```
//služba a charakteristika, skrze kterou bude probíhat komunikace
public static final String CUSTOM_SERVICE = "0000ffe0-0000-1000-8000-
00805f9b34fb";
public static final String CUSTOM_CHARACTERISTIC = "0000ffe1-0000-1000-
8000-00805f9b34fb";
BluetoothGatt;

//proměnná, sloužící ke komunikaci
BluetoothGattCharacteristic char_TX;

//připojení se k zařízení
bluetoothGatt = device.connectGatt(this, false, GATT_Callback);

//získání služeb, které HM-10 modul obsahuje
private final BluetoothGattCallback GATT_Callback = new
BluetoothGattCallback() {
    @Override
    public void onConnectionStateChange(final BluetoothGatt gatt, final int
    status, final int newState) {
        bluetoothGatt.discoverServices();
    }
    @Override
    public void onServicesDiscovered(final BluetoothGatt gatt,
    final intstatus) {
        for(BluetoothGattService gattService : gattServices) {
            String SERVICE_UUID = gattService.getUuid().toString();
            //získání námi hledané služby
            if(SERVICE_UUID.equals(CUSTOM_SERVICE)) {
                System.out.println("Custom Service HM-10Found!");

            //při získání služby je nutné zjistit její charakteristiky
            List<BluetoothGattCharacteristic> gatChars =
            gattService.getCharacteristics();

            for(BluetoothGattCharacteristic gattChar : gattChars) {
                String CHARACTERISTIC_UUID =
                gattCharacteristic.getUuid().toString();
                if(CHARACTERISTIC_UUID.equals(CUSTOM_CHARACTERISTIC)) {
                   System.out.println("Custom Char HM-10 Found! ");
                   //hledaná charakteristika byla nalezena
                   char_TX = gattChar;
                }
            }
        }
    }
}; //GATT_Callback
```

Po tomto procesu získání charakteristiky se uživateli zobrazí část aktivity, která byla do této doby skryta. V této části uživatel vidí jednotlivé položky své vizitky, kterou má uloženou v úložišti mobilního zařízení. Dále má zde volbu upravit data vizitky, nahrát vizitku skrze BLE receiver modul do Arduino zařízení nebo začít s procesem vysílání a skenování vizitek. Dalším důležitým bodem je tedy poslání dat naší vizitky do Arduino zařízení. Tam se data uloží do EEPROM paměti, což ale řeší zdrojový kód na straně Arduina (viz kapitola 6.2). Pro nahrání samotné vizitky slouží tlačítko „Nahrát data“. Tento nahrávací proces je zobrazen následujícím pseudokódem opět s komentáři [26]:

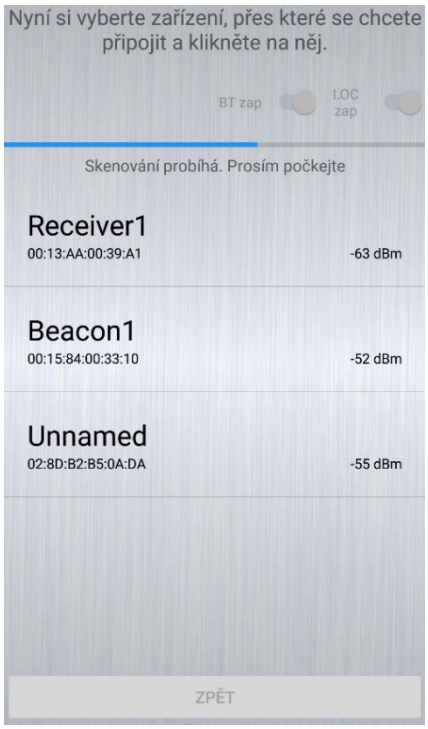
```
//pro Bluetooth je definován maximální počet bytů, které lze přenést v
jednom paketu
public static final int MAX_BYTES_TO_TRANSFER = 20;

//UUID hodnota descriptoru, který se vztahuje k charakteristice
public static final String CLIENT_CHARACTERISTIC_CONFIGURATION = "00002902-
0000-1000-8000-00805f9b34fb";

//po kliknutní na tlačíto poslat data
sendDataBTN.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        sendMyData();
    }
});

//funkce, zajišťující logiku poslání dat
public void sendMyData() {
    String name = myName + ";";
    String age = Integer.toString(myAge) + ";";
    String hobbies = myHobbies;
    //vytvoření jediného Stringu, který obsahuje veškerá data
    String str = name + age + hobbies;

    //nutno zjistit počet paketů, ve kterých budou data obsažena
    int numOfArrays = str.length() / MAX_BYTES_TO_TRANSFER;
    if(str.length() % MAX_BYTES_TO_TRANSFER != 0)
        numOfArrays++;

    //rozdělení celkového řetězce na řetězce o délce 20 bytů
    String[] result = splitStringIntoPackets(str);

    //data se pošlou skrze tuto funkci
    sendPackets(result,numOfArrays);

    //další nastavení vlastností GATT profilu
    bluetoothGatt.setCharacteristicNotification(char_TX,true);
    BluetoothGattDescriptor descriptor = char_TX.getDescriptor(
    UUID.fromString(CLIENT_CHARACTERISTIC_CONFIGURATION));
    descriptor.setValue(BluetoothGattDescriptor.ENABLE_NOTIFICATION_VALUE);
    bluetoothGatt.writeDescriptor(descriptor);
}

public void sendPackets(String[] value, int numberOfPackets) {
    //není potřeba dostávat zpětná upozornění
    char_TX.setWriteType(
    BluetoothGattCharacteristic.WRITE_TYPE_NO_RESPONSE);
    for(int i = 0; i < numberOfPackets; i++) {
```

```
            //odeslání dat po 20ti bytech nastavením hodnoty charakteristiky
            char_TX.setValue(value[i]);
            bluetoothGatt.writeCharacteristic(char_TX);

            //nutná pauza pro synchronizaci s Arduino zdrojovým kódem
            try { Thread.sleep(200); }
            catch (InterruptedException e) { e.printStackTrace(); }
        }
    }
```

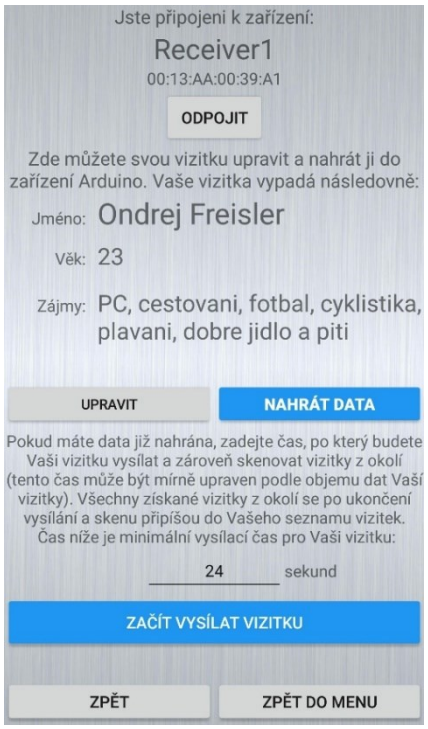
Ke spuštění vysílání vizitky uživatele a skenování dat z okolí slouží část zdrojového kódu, zabývající se mimo jiné výpočtem minimálního času (ten je potřebný pro úspěšné vyslání všech částí (paketů) dat) a veškeré problematiky s tímto spojené. Tento minimální čas, vypočítaný ve vývojovém prostředí Android Studio se ovšem musí dostat do zdrojového kódu Arduina. To je zajištěno opět pomocí nastavení hodnoty charakteristiky, přes kterou komunikujeme. V tomto konkrétním případě s časem je nastavena hodnota charakteristiky na „{POČETSEKUND_“. Když totiž jako vstupní znak na straně zdrojového kódu Arduino zařízení přijde znak „{“, Arduino již ví, že má další znaky ukládat jako hodnotu času do té doby, než přijde vstupní znak „_“. Tento znak zase Arduino rozpozná a vhodně na něj opět zareaguje.

Využití konkrétního znaku pro provedení konkrétní funkce bylo v rámci této práce vytvořeno náhodou. Tyto speciální znaky, na kterých je vlastně postavena logika komunikace, se nesmí vyskytovat jako data samotné vizitky, protože se také nastavují jako hodnota charakteristiky – obdobně jako data vizitky. Funkce, zajišťující zpracování těchto znaků, se vyskytuje jak v Android aplikaci, tak ve zdrojovém kódu Arduino zařízení.

Co se již spuštěného vysílání a skenování vizitek týče, výsledky skenování se připíšou do souboru TempScans.txt, který umožňuje transfer veškerých těchto výsledků mezi aktivitami jednodušší.



Obrázek 16: Aktivita Device_Scanner

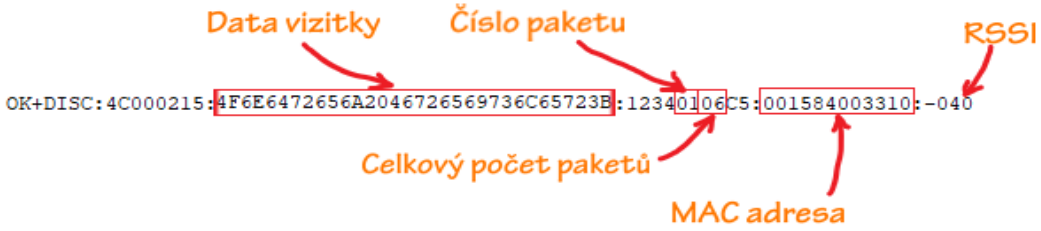


Obrázek 17: Aktivita Connect_Device

Na obrázku 16 (vlevo) je zobrazena aktivita Device_Scanner. Na tomto obrázku je vidět právě probíhající skenování, které prozatím zjistilo 3 BLE zařízení v okolí. Obrázek 17 (vpravo) zase znázorňuje obsah aktivity Connect_Device, kde je znázorněno připojení k BLE zařízení s názvem Receiver1.

6.1.5 Seznam vizitek

Jako poslední část Android aplikace byla vytvořena aktivita s názvem List_of_Scans. Obsah této aktivity je zřejmý již z jejího názvu. Veškerá data, která vytvoří proces skenování vizitek, jsou do této aktivity poslány při jejím spuštění. Formát těchto dat je zobrazen na obrázku 18. O procesu tvorby těchto datových paketů, vysílání a skenování pojednává blíže kapitola 6.2.2.



Obrázek 18: Popis datového paketu, vzniklého skenováním

Tyto data jsou uložená v souboru TempScans.txt a musí být následně zpracovávána. Nejprve se odstraní duplicitní skeny a odstraní se nepotřebné byty. V případě skenu, uvedeného výše, by po odstranění těchto bytů zůstalo toto:

```
4C000215:4F6E6472656A2046726569736C65723B:12340106C5:001584003310
```

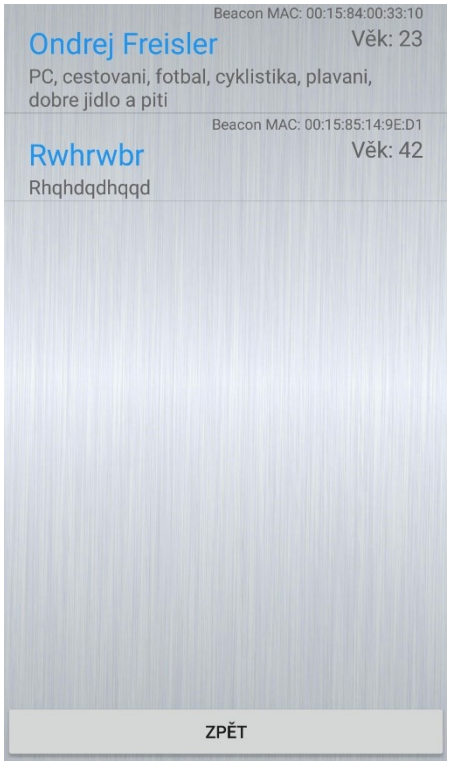
Dále přichází na řadu filtr. Jeho úkolem je zjistit, zda data splňují určitý formát, daný regulárním výrazem. Tento filtr je zde uveden z důvodu, kdy se při skenování mezi daty občas vyskytují nerozpoznatelné znaky, kvůli kterých by následný kód způsobil pád celého programu, neboť počítá se vstupem pouze alfanumerických znaků a dvojtečky. Pokud data filtrem projdou, přichází na řadu proces parsování (syntaktická analýza) dat. Tímto parsováním dostaneme z každého skenu MAC adresu, ze které daný sken pochází a následně odpovídající data. Tato MAC adresa se uloží do souborů ScannedData.txt a později i do MacAddresses.txt, které jsou opět v úložišti mobilního zařízení. Co se týče odpovídajících dat pro každou MAC adresu, ty jsou posléze vyhledávány mezi všemi dostupnými skeny. Toto vyhledávání popisuje následující pseudokód:

```
String customData = ""; //proměnná s později nalezenými daty vizitky
int lookingForPacketNumber = 1; //vždy se začíná paketem číslo 1
int forLoops = 0;

//dokud nebyly nalezeny všechny pakety pro danou MAC adresu nebo dokud
neproběhl cyklus while alespoň 10x
while(foundPackets < numberOfPacketsNewMacHas && forLoops < 10){
    for(int j = 0; j < data.length; j++){

        //hledání čísla následujícího paketu mezi skeny; tento sken musí
        zároveň obsahovat stejnou MAC adresu
        if(data[j].contains(newMAC) && Character.getNumericValue(
        data[j].charAt(38)) == lookingForPacketNumber) {
            //pokud je požadovaný paket nalezen, přidají se jeho data do
```

```
                společného řetězce
                customData += data[j].substring(0,32);
                //zvýšení proměnných int o 1 pro cyklus while
                foundPackets++;
                lookingForPacketNumber++;
            }
        }
        forLoops++;
}

//výsledný řetězec, který se uloží do souboru ScannedData.txt
String result = newMAC+";"+numberOfPacketsNewMacHas+";"+customData+"\n";
//zapsání řetězce, ve kterém jsou zapsána veškerá data, týkající se jedné
konkrétní MAC adresy, do souboru
writeToSDFile(result,"ScannedData.txt", true);
//zapsání dané MAC adresy do souboru, který zajišťuje logiku přidání versus
přepsání vizitky
writeToSDFile(macs[i] + "\n","MacAddresses.txt", true);
```

Pokud nebyly během skenování získány všechny části vizitky z určité MAC adresy (tzn., že jsme získali např. pouze tři pakety z pěti), a to buď z důvodu konce vysílání nebo že během vysílání došlo k výskytu neznámého znaku, zavedením proměnné *forLoops* v cyklu while zabráníme nekonečnému zacyklení celé aplikace.

V opačném případě již zbývá data jen převést do čitelné podoby neboli jinak řečeno zkonvertovat každý byte na znak. K tomuto problému slouží funkce *convertHexToString(String hex)*. Posledním procesem této aktivity je přidání takto zkonvertovaných dat do výsledného seznamu vizitek, který má uživatel možnost již vidět na obrazovce svého mobilního zařízení. Tento seznam funguje na stejném principu a byl vytvořen stejným způsobem, jako se uvádí v kapitole 6.1.3 [19].



Obrázek 19: Aktivita List_of_Scans

6.2 Arduino sketch

Zdrojový kód (sketch), který je nutné nahrát na zařízení Arduino UNO R3, je nezbytný pro funkčnost vzájemné komunikace BLE modulů HM-10 s mobilním zařízením. Sketch byl vyvíjen ve vývojovém prostředí Arduino IDE, o němž již pojednává kapitola 1.4.

Dal by se rozdělit na dvě hlavní části. V první části je řešeno zpracování příchozích bytů, které jsou nastavovány jako hodnota charakteristiky BLE modulu. Druhá část se věnuje samotnému vysílání vizitky uživatele, skenování vizitek z okolí (tzn. z dosahu signálu modulu, který provádí skenování). Zde se také zpracovávají data, uložená v EEPROM paměti, do paketů, které jsou specifické právě pro moduly HM-10. Obě tyto hlavní části se nacházejí ve funkci *loop*, která se volá do doby, než dojde k odpojení napájení celého Arduino zařízení nebo přepsání jiným sketchem. V inicializační funkci *setup* se nastavuje jednotlivým BLE modulům modulační rychlost (měřeném v baudech), která udává počet změn stavu přenosového média za jednu sekundu. Pro komunikaci s BLE moduly byla využita volně dostupná knihovna SoftwareSerial.h, jenž je součástí prostředí Arduino IDE [20].

6.2.1 Zpracování příchozích bytů

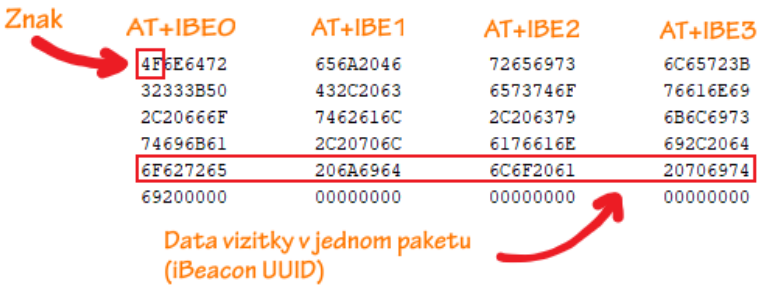
Dokud jsou v sériovém bufferu (ten obsahuje characteristic values již dříve zmíněné Custom Characteristic – viz kapitola 6.1.4) receiver modulu data, ukládají se po jednom bytu do proměnné návratového typu *char*. Každý takto získaný byte prochází funkcí *switch*, která je pro každý znak jakýmsi semaforem a určuje tedy, co se s tímto znakem bude dít dále. Právě zde se vyskytují speciální znaky jako např. „{“ a „_“, které Arduino rozpozná a upraví následně příslušné pomocné proměnné tak, aby *loop* funkce pokračovala tím správným směrem. Pokud je však příchozí byte jinou hodnotou, uloží se do řetězce. Tento řetězec se po vyprázdnění sériového bufferu stává samotnými daty vizitky, která je následně uložena do EEPROM paměti. Pro práci s EEPROM pamětí je využita knihovna EEPROM.h. Po vyprázdnění bufferu, respektive uložení vizitky do EEPROM paměti, je Arduino se všemi připojenými BLE moduly v nečinném stavu a čeká se na přijetí dalšího znaku skrze sériový buffer.

6.2.2 Tvorba paketů

Pokud uživatel klikne v aktivitě Connect_Device v Android aplikaci na tlačítko „Začít vysílat vizitku“, Arduino se o tom dozví opět pomocí speciálního znaku, který je charakteristický pouze pro start této činnosti. Společně se speciálním znakem se dostane do bufferu i čas v sekundách, pomocí kterého dojde k výpočtům potřebných proměnných, na kterých závisí následná tvorba paketů. Ta je řešena v závislosti na tom, co umožňuje modul HM-10. S tímto modulem je možná komunikace pomocí AT příkazů, které mu umožňují chovat se i jako vysílač neboli beacon. Přesněji iBeacon, což je typ vysílačů, vyvinutý společností Apple. Pro tento typ podporují moduly HM-10 sérii příkazů, pomocí nichž je možno nastavit tyto moduly jako iBeacon vysílače. Na obrázku 20 je zobrazeno schéma vysílacího paketu, který využívají právě typy vysílačů iBeacon.

Obrázek 20: Formát rozložení dat v iBeacon vysílačích [21]

V rámci HM-10 modulu je možné nastavit pomocí AT příkazů tři části iBeacon vysílacího paketu: První částí je UUID (příkazy AT+IBE0XXXX až AT+IBE3XXXX), kde za „X“ se dosadí data vizitky v hexadecimálním tvaru – viz obrázek 21, znázorňující data vizitky, rozdělená do paketů, v hexadecimálním tvaru [17].



Obrázek 21: Tvorba advertising paketů

Další části, nastavitelné pomocí AT příkazů, jsou Major Value (AT+MAJR0xXXXX) a Minor Value (AT+MINO0xXXXX). Dva byty Minor Value byly využity pro identifikaci jednotlivých paketů, kdežto byty Major Value nebyly nutné využít. Dále je zásadní nejprve uvést celý modul právě do beacon módu, a to za využití příkazů „AT+ROLE1“ a „AT+IMME1“, které nastaví modul central roli dle GAP profilu. Shrnutím se tedy data vizitky, která program získá z paměti EEPROM, následně nastavují jako součásti AT příkazů [17].

6.2.3 Vysílání a skenování vizitek

Procesy vysílání a skenování vizitek přímo navazují na tvorbu paketů. Jakmile jsou pakety vytvořeny a data vizitky jsou tedy správně rozložena pomocí příkazů „AT+IBEx“, přichází na řadu samotný příkaz, který zapne beacon mód (AT+IBEA1). Ihned po tomto příkazu se HM-10 modul, pojmenovaný Beacon, stane vysílačem. Zpočátku tedy obsahuje první část dat vizitky (první řádek dat z obrázku 21) společně s Minor Value, která identifikuje v tomto případě paket jako první z šesti (6 řádků s daty). Ostatní uživatelé by při spuštění skenování na svém mobilním zařízení již byli schopni tento paket zachytit. Po uplynutí stanoveného času (4 sekundy), což je čas pro vysílání jednoho paketu, se vždy opět volají příkazy „AT+IBEx“ s daty z následujících paketů. Na tomto principu je tedy založeno samotné vysílání [17].

Během každých těchto čtyř sekund je prostor i pro skenování vizitek z okolí. Skenování probíhá na HM-10 modulu, pojmenovaném Scanner. Příkaz, zahajující skenování, se jmenuje „AT+DISI?“. Získání dat (skenů) z BLE vysílačů funguje na obdobném principu jako se zpracovávají příchozí byty (kapitola 6.2.1). S ohledem na dostupnou paměť, kterou Arduino UNO R3 vymezuje pro proměnné, bylo rozhodnuto, že maximální počet zařízení, ze kterých mohou během skenování být data získávána, je 10. Po naplnění těchto maximálně deseti proměnných se jejich obsah ihned (opět z důvodu omezené paměti) posílá skrze receiver modul, který je stále propojený s mobilním zařízením. V něm je totiž podstatně větší prostor pro práci s těmito proměnnými [17].

Jakmile uplyne čas v sekundách, který uživatel v Android aplikaci před začátkem vysílání a skenování zadal, proces vysílání a skenování se automaticky ukončí a uživatel je o této změně stavu v aplikaci upozorněn.

Závěr

Tématem této bakalářské práce bylo vytvoření jednoduchého komunikačního zařízení s využitím technologie Bluetooth 4.0.

Teoretická část práce byla věnována platformě Arduino. V této části byly uvedeny základní informace týkající se platformy Arduino, a to včetně její historie. Následně byla tato platforma podrobena podrobnější analýze a v této kapitole byly také uvedeny informace o jejím zastoupení na trhu. Jako druhým, stěžejním tématem, kterému se teoretická část věnovala, byla bezdrátová komunikační technologie Bluetooth. Obdobně jako u platformy Arduino i v tomto případě byly uvedeny základní informace včetně historie, následované popisem architektury a uvedením důležitých informací, které jsou nezbytné pro řešení této bakalářské práce. Bylo také nutné provést základní analýzu vztahující se k Bluetooth profilům, které jsou úzce spjaty s řešením této BP.

V rámci řešení praktické části, je před jejím samotným vypracováním uveden návrh řešení s příkladem uvedení v praxi. Tento návrh, skládající se ze schématu propojení jednotlivých dílčích zařízení a průběhu komunikace mezi nimi, je znázorněn i graficky a jednotlivé části vývoje a jejich principy jsou v bodech přehledně popsány a vysvětleny. Dalším krokem byl pak vývoj samotného komunikačního zařízení, skládající se ze dvou částí. První část se zaměřuje na mobilní zařízení s operačními systémy Android. V této části bylo nutné vytvořit aplikaci, která má za cíl řídit veškerou probíhající komunikaci mezi uživatelem (jeho mobilním zařízením) a samotným komunikačním zařízením (Arduino zařízení s připojenými Bluetooth moduly) a také zajišťovat uživateli přehled o nashromážděných datech, která uživatel získal v rámci komunikace. Druhá část vývoje se týká vývojového prostředí Arduino IDE, které zajišťuje funkci Bluetooth modulů a vývojové desky Arduino UNO R3. Nakonec byl vývoj vytvořeného komunikačního zařízení zdokumentován.

Bakalářská práce by mohla být dále rozšířena v několika směrech. Jednalo by se například o vyladění veškerých chyb a problémů, které občas vznikají během bezdrátového přenosu dat. Konkrétně se jedná o občasný výskyt náhodných dat na straně Bluetooth skeneru. Další optimalizace by mohla spočívat v podstatném zmenšení rozměrů výsledného zařízení, která by v sobě zahrnovala i vytvoření vlastního plošného spoje a zajišťovala tak uživateli mnohem snadnější přenositelnost celého zařízení. Vhodná by se jevila i možnost využití většího paměťového prostoru na straně Arduino zařízení, která by umožňovala zpracovávat více dat najednou.

Použitá literatura

[1] Arduino – Introduction. [online]. [cit. 2019-04-03]. Dostupné z: https://www.arduino.cc/en/Guide/Introduction

[2] Co je to Arduino? [online]. [cit. 2018-11-03]. Dostupné z: http://czechduino.cz/?co-je-to-arduino,29

[3] Invention Story and History of Development of Arduino. [online]. [cit.2018-11-03]. Dostupné z: http://www.circuitstoday.com/story-and-history-of-development-of-arduino

[4] BARRAGÁN HERNANDO. The Untold History of Arduino. [online]. [cit. 2018-11-03]. Dostupné z: https://arduinohistory.github.io

[5] Arduino: vývojový kit pro hrátky s hardware [online]. [cit. 2018-10-11]. Dostupné z: https://www.root.cz/clanky/arduino-vyvojovy-kit-pro-hratky-s-hardware/

[6] Arduino – Products. [online]. [cit. 2019-04-03]. Dostupné z: https://www.arduino.cc/en/Main/Products

[7] VODA ZBYŠEK. Arduino klony – Co to je? Koupit či nekoupit? [online]. 12. 6. 2017 [cit. 2018-10-11]. Dostupné z: https://arduino.cz/arduino-klony-co-to-je-koupit-ci-nekoupit/

[8] Arduino Board Description. [online]. [cit. 2018-10-11] Dostupné z: https://www.tutorialspoint.com/arduino/arduino_board_description.htm

[9] Arduino Playground – What Adapter. [online]. [cit. 2018-10-11]. Dostupné z: https://playground.arduino.cc/Learning/WhatAdapter

[10] Quora - What is the function of ICSP pins on the Arduino UNO? [online]. [cit. 2018-10-11]. Dostupné z: https://www.quora.com/What-is-the-function-of-ICSP-pins-on-the-Arduino-Uno

[11] ALLAN A., COLEMAN D., MISTRY S. Make: Bluetooth: Bluetooth LE Projects with Arduino, Raspberry Pi, and Smartphones. Maker Media 2016. 238 stran. ISBN 978-1-457-18709-4

[12] ŠEBESTA ROMAN, DVORSKÝ MAREK. Rádiové sítě I pro integrovanou výuku. [online]. 2014 [cit. 2018-10-11]. Dostupné z: https://lms.vsb.cz/pluginfile.php/647225/mod_resource/content/1/rs1_141022_skripta.pdf

[13] ŠKOPEK PAVEL. Techbox: Bluetooth sjednotilo bezdrátovou komunikaci. [online]. 24. 5. 2013 [cit. 2018-10-11]. Dostupné z: https://mobilenet.cz/clanky/techbox-bluetooth-sjednotilo-bezdratovou-komunikaci-12085

[14] GRADY SARA. Origin Stories: Bluetooth | Helix Magazine. [online]. 30. 6. 2017 [cit. 2019-03-04]. Dostupné z: https://helix.northwestern.edu/blog/2017/06/origin-stories-bluetooth

[15] 6. Bluetooth Low Energy – TUTORIALS 0.0.1 documentation. [online]. [cit. 2019-03-04]. Dostupné z: https://mbientlab.com/tutorials/AboutWireless.html

[16] KILIÁN KAREL. Bluetooth 5: jaké jsou největší výhody proti starší verzi 4.2? [online]. 20. 12. 2018 [cit. 2019-03-04]. Dostupné z: https://www.svetandroida.cz/bluetooth-5/

[17] HM-10 DataSheet. [online]. Dostupné z: https://people.ece.cornell.edu/land/courses/ece4760/PIC32/uart/HM10/DSD%20TECH%20HM-10%20datasheet.pdf

[18] dBm to mW conversion calculator. [online]. [cit. 2019-04-03]. Dostupné z: https://www.rapidtables.com/convert/power/dBm_to_mW.html

[19] KIM YONG MOOK. How to convert Hex to ASCII in Java. [online]. 20. 1. 2010, poslední revize 30. 8. 2012 [cit. 2019-04-03]. Dostupné z: https://www.mkyong.com/java/how-to-convert-hex-to-ascii-in-java/

[20] Baud – Wikipedie. [online]. poslední revize 5. 2. 2014 [cit. 2019-04-03]. Dostupné z: https://cs.wikipedia.org/wiki/Baud

[21] BLE Eddystone beacon vs iBeacon? Beacon Communication basics. [online]. Dostupné z: https://www.beaconstac.com/ibeacon-and-eddystone

[22] File: Arduino Logo.svg - Wikimedia Commons. [online]. Dostupné z: https://commons.wikimedia.org/wiki/File:Arduino_Logo.svg

[23] Android Ios HM-10 Ble Bluetooth 4.0 Cc2540. [online]. Dostupné z: https://www.daraz.pk/products/android-ios-hm-10-ble-bluetooth-40-cc2540-i1849099-s9794081.html

[24] PRATAMA MUHAMMAD RIDHO K. Implementasi CoreBluetooth untuk Komunikasi antara iOS dan BLE Peripheral. [online]. 18. 11. 2017. Dostupné z: https://medium.com/99ridho/implementasi-corebluetooth-untuk-komunikasi-antara-ios-dan-ble-peripheral-1dfa7ae2c1f

[25] CHEN DAN. How to Use Bluetooth 4.0 HM10 : 4 Steps (with Pictures). [online]. Dostupné z: https://www.instructables.com/id/How-to-Use-Bluetooth-40-HM10/

[26] Bluetooth low energy overview | Android Developers. [online]. [cit. 2019-04-09]. Dostupné z: https://developer.android.com/guide/topics/connectivity/bluetooth-le#permissions

[27] Introduction to Activities | Android Developers [online]. [cit. 2019-04-09]. Dostupné z: https://developer.android.com/guide/components/activities/intro-activities

[28] Actionable Insights: How data presentation skills create value – Annalect. [online]. Dostupné z: https://www.annalect.fi/actionable-insights-data-presentation/

[29] Arduino – Home. [online]. Dostupné z: https://www.arduino.cc/

[30] Interaction Design Institute Ivrea. [online]. Dostupné z: http://interactionivrea.org/en/index.asp

Seznam příloh

Příloha A: *USB flash disk*

Obsah:

- Adresář s aplikací, vyvíjenou v programu Android Studio 3.3.1,
- Arduino sketch, vytvořený v programu Arduino IDE 1.8.9